DATA BOOK

µ

SOFT
MICROCONTROLLER

DALLAS
SEMICONDUCTOR

TABLE OF CONTENTS

SOFT MICROCONTROLLER USER'S GUIDE

Section 1 Introduction 1
Section 2 Selection Guide 4
Section 3 Soft Microcontroller Architecture 6
Section 4 Programmer's Guide 10
Section 5 Memory Interconnect 49
Section 6 Lithium Backup 56
Section 7 Power Management 61
Section 8 Software Control 66
Section 9 Firmware Security 73
Section 10 Reset Conditions 83
Section 11 Interrupts 90
Section 12 Parallel I/O 97
Section 13 Programmable Timers 106
Section 14 Serial I/O 111
Section 15 Cpu Timing 125
Section 16 Program Loading 131
Section 17 Real–time Clock 146
Section 18 Troubleshooting 166
Section 19 Instruction Set Details 169

SOFT MICROCONTROLLER FAMILY DATA SHEETS

DS2250(T) Soft Microcontroller 176
DS2251(T) 128K Soft Microcontroller 195
DS2252(T) Secure Microcontroller 215
DS5000(T) Soft Microcontroller 229
DS5000FP Soft Microcontroller Chip 247
DS5001FP 128K Soft Micro Chip 267
DS5002FP Secure Micro 290

DEVELOPMENT TOOLS

Development Support 317
DS5000TK Evaluation Kit 323

TABLE OF CONTENTS

SOFT MICROCONTROLLER USER'S GUIDE

Section 1 Introduction 1
Section 2 Selection Guide 4
Section 3 Soft Microcontroller Architecture 6
Section 4 Programmer's Guide 10
Section 5 Memory Interconnect 49
Section 6 Uptime Backup 58
Section 7 Power Management 61
Section 8 Software Control 68
Section 9 Firmware Security 79
Section 10 Reset Conditions 82
Section 11 Interrupts 90
Section 12 Parallel I/O 97
Section 13 Programmable Timers 106
Section 14 Serial I/O 111
Section 15 Cpu Timing 125
Section 16 Program Loading 131
Section 17 Real-time Clock 146
Section 18 Troubleshooting 156
Section 19 Instruction Set Details 160

SOFT MICROCONTROLLER FAMILY DATA SHEETS

DS2250(T) Soft Microcontroller 176
DS2251(T) 128K Soft Microcontroller 190
DS2252(T) Secure Microcontroller 215
DS5000(T) Soft Microcontroller 229
DS5000FP Soft Microcontroller Chip 247
DS5001FP 128K Soft Micro Chip 267
DS5002FP Secure Micro 290

DEVELOPMENT TOOLS

Development Support 317
DS5000TK Evaluation Kit 329

SOFT MICROCONTROLLER
USER'S GUIDE

SOFT MICROCONTROLLER
USER'S GUIDE

SECTION 1: INTRODUCTION

The Soft Micro family is a line of 8051 compatible microcontrollers that are based on nonvolatile RAM rather than ROM. Using NVRAM in a micro provides several features that system designers need, but that have never been available in an off–the–shelf microcontroller. The line is divided between chips and modules. The chips are monolithic microcontrollers that connect to standard SRAM and a lithium battery. Modules combine the chips with the RAM and lithium cell. All parts in the Soft Micro line provide the following basic attributes:

- Fully code and resource compatible with the 8051
- Memory access on a separate bus, preserving I/O ports
- Control (chip) or incorporate (module) a large nonvolatile memory
- 10–year data retention
- In–system program loading via serial port
- Crashproof Operation
  - Power–Fail Reset
  - Early Warning Power Fail Interrupt
  - Watchdog Timer

SEPARATE ADDRESS/DATA BUS

Soft Micro chips provide a non–multiplexed address/data bus that interfaces to memory without interfering with I/O ports. This Byte–wide bus connects directly to standard CMOS SRAM in 8K x 8, 32K x 8, or 128K x 8 densities with no glue logic. Note that this is in addition to the standard 8051 port 0 and 2 multiplexed bus. In module form, the Byte–wide bus is already connected directly to on–board SRAM, so the memory access becomes transparent and the I/O ports free for application use. The extra memory bus also allows for a time–of–day function to be included, and all Soft Micro modules are available with built in real–time clocks. The same clock devices are individually available when building a system from chips. Battery backup and decoding are automatically handled by the microcontroller.

LARGE NONVOLATILE MEMORY

Soft Micro chips provide nonvolatile memory control for standard CMOS SRAM. Modules combine the micro chip with memory and lithium backup. This includes conditionally write protected chip enables and a power supply output that switches between +5V and battery backup. The chip enables are decoded based on user selections. The micro decodes chip enables based on user selections of memory size and partitioning. Partitioning defines the portion of memory used for program and data segments. Areas that are designated program are always write protected and are treated as ROM. Data areas are write protected only when power is out of tolerance. Users select the desired partition and the microcontroller handles the memory decoding automatically. A large nonvolatile memory is useful for data logging and as flexible program storage. Memory will be retained for over 10 years in the absence of power by ultra low–leakage lithium backed circuits.

IN–SYSTEM LOADING

Combining the NVRAM with the in–system programming capability lets the user update program code at any time. This program loading is supervised by a built–in ROM–based bootstrap loader. The ROM loader becomes transparent once program loading is complete. All Soft Micro devices allow program loading via the micro serial port. Data memory can also be retrieved using this loader function. Selected versions provide other parallel loading protocols as well. In–system loading allows a system to be configured during final system test. A user can load custom software, diagnostic routines, or calibration constants. If something changes or new features arise, the system can then be reprogrammed while in the field.

CRASHPROOF OPERATION

Soft Micro devices are designed for unsupervised operation in remote locations. The crashproof features prevent a micro from running out of control during transient events. Features that contribute to crashproof operation include a reset when power is out of tolerance; an early warning power–fail interrupt that allows software to save critical data; and a watchdog to reset the micro if it gets lost. Also, nonvolatile memory allows software to save the operating state so a task can be resumed when power returns to normal.

SOFTWARE SECURITY

An important feature on Soft Micro family devices is firmware or memory security. The original DS5000 provides extensive means to secure the memory contents. This prevents unauthorized copying of firmware and prevents access to critical data values. Also, RAM is inherently more secure than ROM since it can be repro-

grammed or destroyed if necessary. The DS5002 is a second generation with even more security features. The DS5000 provides optional security. That is, a user must enable the function. The DS5002 provides security all the time. Security features include real–time high speed memory encryption, false generation of dummy addresses on the bus, and a self–destruct input for tamper protection.

The Soft Microcontroller family consists of three chips and their associated modules. Each chip provides most of the Soft Micro features discussed below. Differences stem from I/O, memory access, and security features. The DS5000FP is used in DS2250(T) and DS5000(T) modules. The DS5001FP is used in the DS2251(T), and the DS5002FP is used in the DS2252(T). A full selector guide with all memory and speed permutations is provided in the next section. A short description of each version is given below.

CHIP	DESCRIPTION	BYTE–WIDE BUS MEMORY ACCESS	SECURITY	PACKAGE
DS5000FP	Soft Micro Chip	8, 32, 64K bytes*	Optional	80–pin QFP
DS5001FP	128K Micro Chip	32, 64, 128K bytes	None	80–pin QFP
DS5002FP	Secure Micro Chip	32, 64, 128K bytes	Maximum	80–pin QFP

MODULE	DESCRIPTION	ON–BOARD MEMORY	PACKAGE
DS2250(T)	DS5000FP on SIMM	8, 32, 64K bytes*	40–pin SIMM
DS5000(T)	DS5000FP in DIP Module	8, 32K bytes	40–pin DIP
DS2251(T)	DS5001FP on SIMM	32, 64, 128K bytes	72–pin SIMM
DS2252(T)	DS5002FP on SIMM	32, 64, 128K bytes	40–pin SIMM

*32K partitionable, 32K restricted to data memory only.

NOTES:
"T" specifies optional on–board real–time clock.
128K byte versions provide 64K program, 64K data. Other versions are partitionable

PRODUCT DESCRIPTION

All devices listed below have the standard 8051 family feature set listed once here for convenience, but not repeated for each device.

- 8051–compatible instruction set
- Addresses 64K program and 64K data memory
- 4 8–bit pseudo–bidirectional I/O ports
- 128 bytes scratchpad RAM
- 2 16–bit timer/counters
- 1 UART
- 5 Interrupts with 2 external

DS5000FP Soft Micro Chip

The DS5000FP is the original Soft Micro chip. It adds the following features to the 8051 set :

- Non–multiplexed Byte–wide address/data bus for memory access.
- Nonvolatile Control for 8K x 8 or 32K x 8 SRAMs
- Partitions one SRAM into program and data areas, and write protects the program segment
- Decodes memory for up to two 32K x 8 SRAMs (# 2 is data memory only)
- Power–fail Reset, and Interrupt
- Precision Watchdog Timer

- ROM based Serial Bootstrap Loader
- Optional security features
  - Memory encryption in real–time
  - 48–bit user selected encryption key
  - Security lock destroys memory if unlocked
  - Vector RAM hides 48 bytes on–chip
  - Dummy operations on the memory bus

DS5000(T) Soft Micro

The DS5000 incorporates the DS5000FP chip in a 40–pin module with an 8051 footprint and pinout.

- Familiar 40–pin DIP package
- Built–in NVRAM of 8K x 8 or 32K x 8
- I/O ports not disturbed by on–board memory access
- 10–year data retention in the absence of power
- Partitions memory into program and data areas, write protects the program segment
- Power–fail Reset and Interrupt
- Precision Watchdog Timer
- ROM based Serial Bootstrap Loader
- Optional memory security
- Optional built–in real–time clock

DS2250(T) Soft Micro Stik

The DS2250 incorporates the DS5000FP chip on a 40–pin SIMM module. It has the identical feature set as the DS5000, but is in a different form–factor. This package change allows up to 64K bytes NVRAM instead of 32K bytes. Note that as mentioned above, the second 32K is restricted to data memory. Like the DS5000, this module guarantees better than 10–year data retention at room temperature.

DS5001FP 128K Micro Chip

The DS5001 provides the base feature set of the DS5000FP with the following extras:

- Accesses up to 128K bytes on the Byte–wide bus.
- Decodes memory for 32K x 8 or 128K x 8 SRAMs.
- 4 additional decoded peripheral chip enables
- CRC hardware for checking memory validity
- Optionally emulates an 8042 style slave interface
- Bandgap reference for more accurate power monitor

Note: The DS5001FP has no memory encryption feature.

DS2251(T) 128K Micro Stik

The DS2251 is a SIMM based on the DS5001. It provides up to 128K bytes of on–board NVRAM and has the Byte–wide bus available at the connector. This is used with the decoded peripheral enables for memory mapped peripherals such as a UART or A/D converter. The real–time clock option is a parallel access type with interrupt capability. Like the older versions, the DS2251 provides 10–year data retention, even in the largest memory configuration.

DS5002FP Secure Micro Chip

The DS5002FP is a highly secure version of the DS5001FP. It provides the operating features of the DS5001, with the following enhancements to the DS5000 security features.

- Security is active at all times
- Improved memory encryption using a 64–bit encryption key
- Automatic random generation of encryption keys
- Self–destruct input for tamper protection
- Die top–coating prevents micro probe

DS2252(T) Secure Micro Stik

The DS2252 incorporates the DS5002FP on a 40–pin SIMM. This includes from 32K bytes to 128K bytes of secure memory with an optional real–time clock. The memory is highly secure from tampering and from competitors. Like other products in the family, the D2252 has a data retention period of over 10–years.

SECTION 2: SELECTION GUIDE

The following configurations are available. Standard versions that are commonly in stock are shown in bold type. Other combinations are available but may not be in stock at all times. Speeds are rated maximums, but the micros are fully static and can be run as slow as desired.

CHIP	DESCRIPTION	MAXIMUM SPEED OPTIONS	PART NUMBER
DS5000FP	**Soft Micro Chip**	**8 MHz**	**DS5000FP-8**
DS5000FP	**Soft Micro Chip**	**12 MHz**	**DS5000FP-12**
DS5000FP	**Soft Micro Chip**	**16 MHz**	**DS5000FP-16**
DS5001FP	**128K Micro Chip**	**12 MHz**	**DS5001FP-12**
DS5001FP	**128K Micro Chip**	**16 MHz**	**DS5001FP-16**
DS5002FP	**Secure Micro Chip**	**12 MHz**	**DS5002FP-12**
DS5002FP	**Secure Micro Chip**	**16 MHz**	**DS5002FP-16**

MODULE	DESCRIPTION	MEMORY	SPEED	CLOCK	PART NUMBER
DS5000	Soft Micro	8K bytes	8 MHz	no	DS5000–08–08
DS5000	Soft Micro	8K bytes	12 MHz	no	DS5000–08–12
DS5000	Soft Micro	8K bytes	16 MHz	no	DS5000–08–16
DS5000	**Soft Micro**	**32K bytes**	**8 MHz**	**no**	**DS5000-32-08**
DS5000	**Soft Micro**	**32K bytes**	**12 MHz**	**no**	**DS5000-32-12**
DS5000	**Soft Micro**	**32K bytes**	**16 MHz**	**no**	**DS5000-32-16**
DS5000T	Soft Micro	8K bytes	8 MHz	yes	DS5000T–08–08
DS5000T	Soft Micro	8K bytes	12 MHz	yes	DS5000T–08–12
DS5000T	Soft Micro	8K bytes	16 MHz	yes	DS5000T–08–16
DS5000T	**Soft Micro**	**32K bytes**	**8 MHz**	**yes**	**DS5000T-32-08**
DS5000T	**Soft Micro**	**32K bytes**	**12 MHz**	**yes**	**DS5000T-32-12**
DS5000T	**Soft Micro**	**32K bytes**	**16 MHz**	**yes**	**DS5000T-32-16**
DS2250	Soft Micro Stik	8K bytes	8 MHz	no	DS2250–08–08
DS2250	Soft Micro Stik	8K bytes	12 MHz	no	DS2250–08–12
DS2250	Soft Micro Stik	8K bytes	16 MHz	no	DS2250–08–16
DS2250	**Soft Micro Stik**	**32K bytes**	**8 MHz**	**no**	**DS2250-32-08**
DS2250	**Soft Micro Stik**	**32K bytes**	**12 MHz**	**no**	**DS2250-32-12**
DS2250	**Soft Micro Stik**	**32K bytes**	**16 MHz**	**no**	**DS2250-32-16**
DS2250	Soft Micro Stik	64K bytes	8 MHz	no	DS2250–64–08
DS2250	Soft Micro Stik	64K bytes	12 MHz	no	DS2250–64–12
DS2250	Soft Micro Stik	64K bytes	16 MHz	no	DS2250–64–16
DS2250T	Soft Micro Stik	8K bytes	8 MHz	yes	DS2250T–08–08

MODULE	DESCRIPTION	MEMORY	SPEED	CLOCK	PART NUMBER
DS2250T	Soft Micro Stik	8K bytes	12 MHz	yes	DS2250T–08–12
DS2250T	Soft Micro Stik	8K bytes	16 MHz	yes	DS2250T–08–16
DS2250T	**Soft Micro Stik**	**32K bytes**	**8 MHz**	**yes**	**DS2250T-32-08**
DS2250T	**Soft Micro Stik**	**32K bytes**	**12 MHz**	**yes**	**DS2250T-32-12**
DS2250T	**Soft Micro Stik**	**32K bytes**	**16 MHz**	**yes**	**DS2250T-32-16**
DS2250T	Soft Micro Stik	64K bytes	8 MHz	yes	DS2250T–64–08
DS2250T	Soft Micro Stik	64K bytes	12 MHz	yes	DS2250T–64–12
DS2250T	Soft Micro Stik	64K bytes	16 MHz	yes	DS2250T–64–16
DS2251	128K Micro Stik	32K bytes	12 MHz	no	DS2251–32–12
DS2251	128K Micro Stik	32K bytes	16 MHz	no	DS2251–32–16
DS2251	**128K Micro Stik**	**64K bytes**	**12 MHz**	**no**	**DS2251-64-12**
DS2251	**128K Micro Stik**	**64K bytes**	**16 MHz**	**no**	**DS2251-64-16**
DS2251	**128K Micro Stik**	**128K bytes**	**12 MHz**	**no**	**DS2251-128-12**
DS2251	**128K Micro Stik**	**128K bytes**	**16 MHz**	**no**	**DS2251-128-16**
DS2251T	128K Micro Stik	32K bytes	12 MHz	yes	DS2251T–32–12
DS2251T	128K Micro Stik	32K bytes	16 MHz	yes	DS2251T–32–16
DS2251T	**128K Micro Stik**	**64K bytes**	**12 MHz**	**yes**	**DS2251T-64-12**
DS2251T	**128K Micro Stik**	**64K bytes**	**16 MHz**	**yes**	**DS2251T-64-16**
DS2251T	**128K Micro Stik**	**128K bytes**	**12 MHz**	**yes**	**DS2251T-128-12**
DS2251T	**128K Micro Stik**	**128K bytes**	**16 MHz**	**yes**	**DS2251T-128-16**
DS2252	**Secure Micro Stik**	**32K bytes**	**12 MHz**	**no**	**DS2252-32-12**
DS2252	Secure Micro Stik	32K bytes	16 MHz	no	DS2252–32–16
DS2252	Secure Micro Stik	64K bytes	12 MHz	no	DS2252–64–12
DS2252	Secure Micro Stik	64K bytes	16 MHz	no	DS2252–64–16
DS2252	Secure Micro Stik	128K bytes	12 MHz	no	DS2252–128–12
DS2252	Secure Micro Stik	128K bytes	16 MHz	no	DS2252–128–16
DS2252T	**Secure Micro Stik**	**32K bytes**	**12 MHz**	**yes**	**DS2252T-32-12**
DS2252T	Secure Micro Stik	32K bytes	16 MHz	yes	DS2252T–32–16
DS2252T	Secure Micro Stik	64K bytes	12 MHz	yes	DS2252T–64–12
DS2252T	Secure Micro Stik	64K bytes	16 MHz	yes	DS2252T–64–16
DS2252T	Secure Micro Stik	128K bytes	12 MHz	yes	DS2252T–128–12
DS2252T	Secure Micro Stik	128K bytes	16 MHz	yes	DS2252T–128–16

SECTION 3: SOFT MICROCONTROLLER ARCHITECTURE

Introduction

The Soft Microcontroller family is based on an 8051 compatible core with a memory interface and I/O logic build around it. Many functions of the Soft Micro are identical to standard 8051s and are documented here for completeness. In general, most architecture features apply to all Soft Micro devices. When there is a difference between versions, this will be mentioned. A block diagram of the microcontroller core is shown in Figure 3–1 below.

Bus Organization

There are four major busses implemented within the Soft Micro that are relevant in developing an understanding of the operation of the device. These are the Internal Data Bus, the Internal Address Bus, the Byte–wide Memory Bus, and the Expanded Bus. All addresses and data which are transferred during program execution are passed on the Internal Address and Data Busses. User Program and Data Memory is always accessed from either the byte–wide Program/Data RAM or from external memory located on the Expanded Bus.

The Soft Microcontroller uses the Byte–wide Memory Bus for access to Program/Data RAM in the same fashion as an 8051 Family device would access internal ROM or EPROM memory. This bus is physically separate from the Expanded Bus which, if used, replaces Port 2 and Port 0 pins.

CPU Registers

All of the CPU registers are mapped as Special Function Registers (SFR's) and are identical in number and function to those present within the 8051. These registers are described briefly below:

Accumulator

The Accumulator (A) is used as either a source or destination register in all arithmetic instructions. It may also be used in most other types of instructions.

Stack Pointer

The Stack Pointer (SP) is an 8-bit register which is used to mark the location of the last byte of data stored in the stack. The stack itself may be located anywhere in the on-chip 128-byte Scratchpad register area. The Stack Pointer pre-increments during a stack push and post-decrements during a stack pop.

B Register

The major function of the B register is as a source and destination register during multiply and divide instructions. It may also be used as a scratchpad register.

Program Status Word

The Program Status Word (PSW) contains status flags that are set according to the results of a previously executed instruction. In addition, the PSW contains register bank select bits.

Data Pointer

The Data Pointer (DPTR) is used to access Data Memory that may be mapped into Byte–wide Data RAM or onto external memory devices on the Expanded Bus. It is accessed by the user's program as either two 8-bit Special Function registers or as a 16-bit register with certain instructions.

Scratchpad Registers

Scratchpad registers are 128 registers where data may be stored directly. They are addressed from 00H to 7FH and may be accessed by a MOV instruction. Included in the scratchpad area are four 8-byte banks of working registers. These registers are not part of the data memory map.

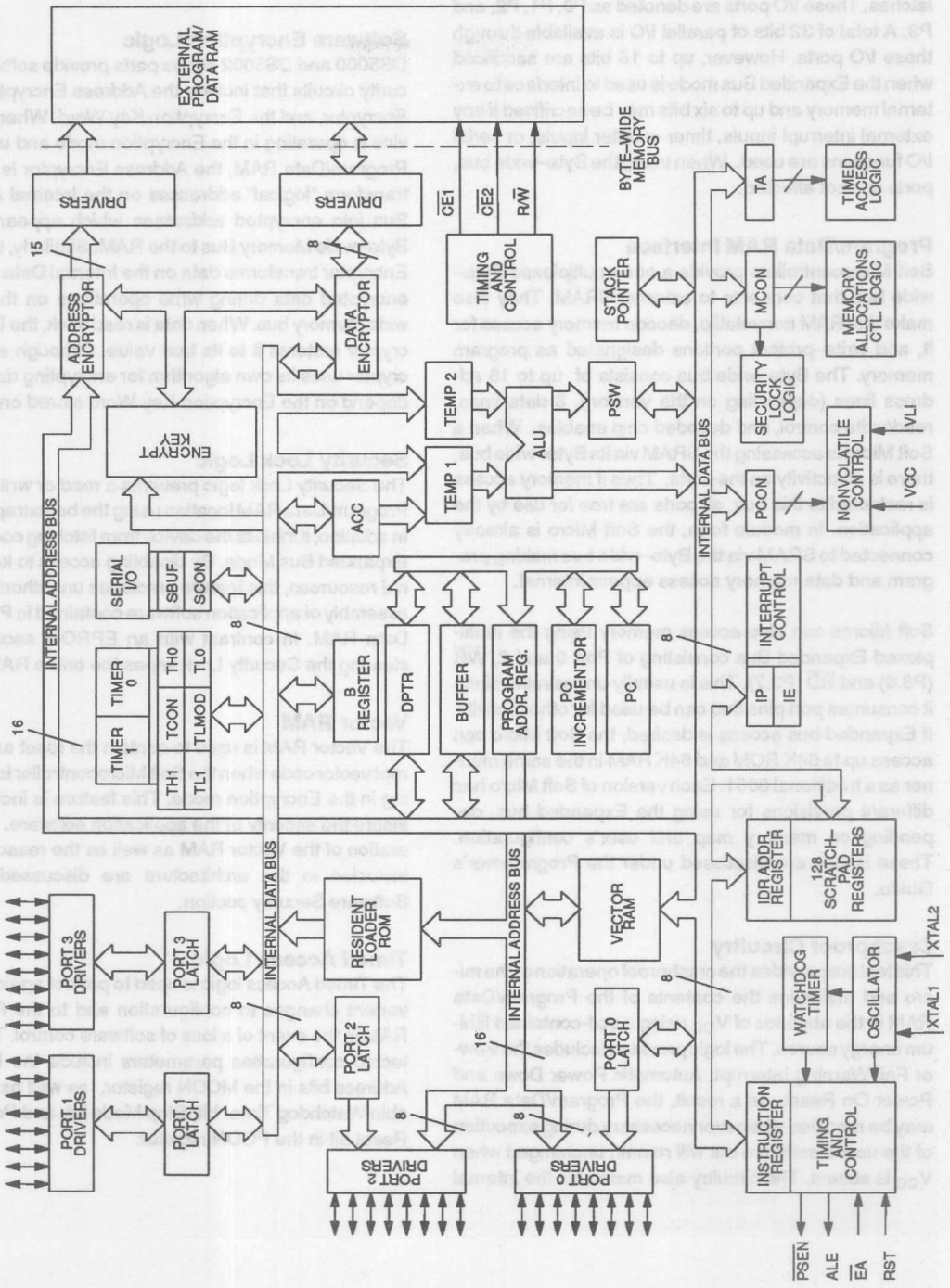
Serial I/O

The on-chip serial I/O port is comprised of a receive data buffer, a transmit data buffer, and a control register. Both the receive data buffer and the transmit data buffer are accessed in a single location (SBUF) in the Special Function Register map. The control register (SCON) is accessed in an separate location. When the serial I/O function is enabled, two external I/O pins (P3.0, P3.1) are re-assigned in hardware to serve the transmit and receive data functions.

Programmable Timers

Two 16-bit programmable timers are included that can perform various timing and counting functions. A total of four registers (TH1, TL1, TH0, and TL0) access the upper and lower halves of each of the two timer/counters. A single control register (TCON) is used to select the various operating modes of the two timers. Two external I/O pins (P3.4, P3.5) may be programmed to serve as external counter inputs, one pin for each of the two timer/counters.

SOFT MICROCONTROLLER ARCHITECTURAL BLOCK DIAGRAM Figure 3–1



Parallel I/O

Four SFR's provide access for the four parallel I/O port latches. These I/O ports are denoted as P0, P1, P2, and P3. A total of 32 bits of parallel I/O is available through these I/O ports. However, up to 16 bits are sacrificed when the Expanded Bus mode is used to interface to external memory and up to six bits may be sacrificed if any external interrupt inputs, timer counter inputs, or serial I/O functions are used. When using the Byte–wide bus, ports are not affected.

Program/Data RAM Interface

Soft Microcontrollers provide a non–multiplexed Byte–wide bus that connects to external SRAM. They also make this RAM nonvolatile, decode memory access for it, and write–protect portions designated as program memory. The Byte–wide bus consists of up to 16 address lines (depending on the version), 8 data lines, read/write control, and decoded chip enables. When a Soft Micro is accessing the SRAM via its Byte–wide bus, there is no activity on the ports. Thus if memory access is restricted to this bus, all ports are free for use by the application. In module form, the Soft Micro is already connected to SRAM via the Byte–wide bus making program and data memory access appear internal.

Soft Micros can also access memory using the multiplexed Expanded Bus consisting of Port 0 and 2, $\overline{WR}$ (P3.6) and $\overline{RD}$ (P3.7). This is usually undesirable since it consumes port pins that can be used for other activity. If Expanded bus access is desired, the Soft Micro can access up to 64K ROM and 64K RAM in the same manner as a traditional 8051. Each version of Soft Micro has different provisions for using the Expanded bus, depending on memory map and user's configuration. These issues are discussed under the Programmer's Guide.

Crashproof Circuitry

This feature provides the crashproof operation of the micro and maintains the contents of the Program/Data RAM in the absence of V_{CC} using a self-contained lithium energy source. The logic provided includes the Power Fail Warning Interrupt, Automatic Power Down and Power On Reset. As a result, the Program/Data RAM may be modified whenever necessary during execution of the user's software but will remain unchanged when V_{CC} is absent. The circuitry also maintains the Internal Scratchpad RAM and certain Special Function registers during a power down condition.

Software Encryption Logic

DS5000 and DS5002 series parts provide software security circuits that include the Address Encryptor, Data Encryptor, and the Encryption Key Word. When the device is operating in the Encryption mode and using the Program/Data RAM, the Address Encryptor is used to transform "logical" addresses on the Internal Address Bus into encrypted addresses which appear on the Byte–wide Memory Bus to the RAM. Similarly, the Data Encryptor transforms data on the Internal Data bus into encrypted data during write operations on the Byte–wide Memory bus. When data is read back, the Data Encryptor restores it to its true value. Although each encryptor uses its own algorithm for encrypting data, both depend on the Encryption Key Word stored on–chip.

Security Lock Logic

The Security Lock logic prevents a read or write to any Program/Data RAM location using the bootstrap loader. In addition, it inhibits the device from fetching code in the Expanded Bus Mode. By disabling access to key internal resources, this feature precludes unauthorized disassembly of application software contained in Program/Data RAM. In contrast with an EPROM security bit, clearing the Security Lock wipes the entire RAM area.

Vector RAM

The Vector RAM is used to contain the reset and interrupt vector code when the Soft Microcontroller is operating in the Encryption mode. This feature is included to insure the security of the application software. The operation of the Vector RAM as well as the reason for its inclusion in the architecture are discussed in the Software Security section.

Timed Access Logic

The Timed Access logic is used to protect against inadvertent changes to configuration and to the Program RAM in the event of a loss of software control. The protected configuration parameters include the Partition Address bits in the MCON register, as well as the Enable Watchdog Timer bit, Stop Mode bit, and Power On Reset bit in the PCON register.

Watchdog Timer

When the user's software is being executed, the Watchdog Timer can be used to automatically restart the processor in the event that software control is lost. It is also used to generate a delay during an oscillator start-up condition which allows the clock frequency to stabilize. This occurs during reset cycles that follow a time in which the oscillator has been stopped (Stop Mode Reset and Power On Reset).

Resident Loader ROM

The Resident Loader ROM contains firmware that controls the initial loading of the nonvolatile Program/Data RAM. The firmware provides Serial Bootstrap Load operation via the on-chip serial port. The internal ROM is not accessible by the user and performs the loading function only when the device is strapped for operation in the Program mode. The ROM becomes transparent to the user once loading is done and has no effect on the memory map.

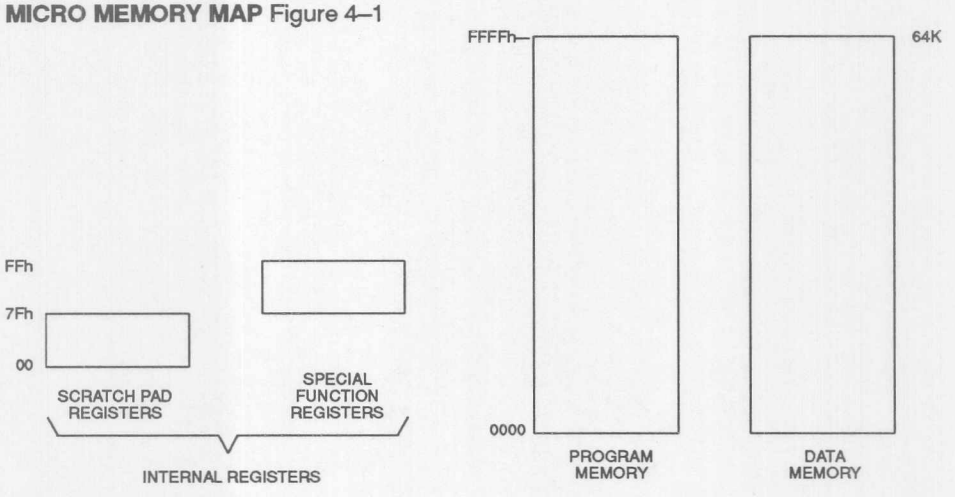
SECTION 4: PROGRAMMER'S GUIDE

The Soft Micro uses nonvolatile RAM technology for both Program and Data memory. It uses NVSRAM in place of ROM by write protecting and decoding memory segments that a user designates as Program memory. The remaining RAM area is used as nonvolatile data storage. One of the advantages of breaking a common RAM into two segments is that a smaller number of memory chips is needed. For example, if a system requires 24K bytes of program memory and 4K bytes of data memory, this all fits within one 32K x 8 SRAM. The Soft Micro can break this RAM into program and data segments, unconditionally write protecting the program area. The process of dividing the common memory space into ROM and RAM is called partitioning. All Soft Micros are capable of doing this. However, there are differences between original DS5000 series [includes DS5000FP, DS5000(T), and DS2250(T)] and newer DS5001 series [includes DS5001FP, DS2251(T), DS5002FP, DS2252(T)]. The original DS5000 series could partition one SRAM of up to 32K bytes. It could access a second RAM, but this was restricted to data memory only. The DS5001 series can partition two 32K byte SRAMs, or even one 128K x 8 SRAM. Common elements of the programming model are given below, with individual differences highlighted.

Soft Micro Memory Organization

All Soft Micros follow the standard 8051 convention of three memory areas. These include Internal registers, Program memory and Data memory. These memory areas are not contiguous and are accessed in different ways. The Soft Micro duplicates all standard 8051 registers and adds several new ones. Soft Micros have a 64K byte program and 64K byte data space. However, the Soft Micros provide several ways to access these areas, and these features are what make the family unique. Figure 4–1 shows the memory map of Soft Micros in general terms. The specific details and access to the memory areas are discussed below.

SOFT MICRO MEMORY MAP Figure 4–1



Internal Registers

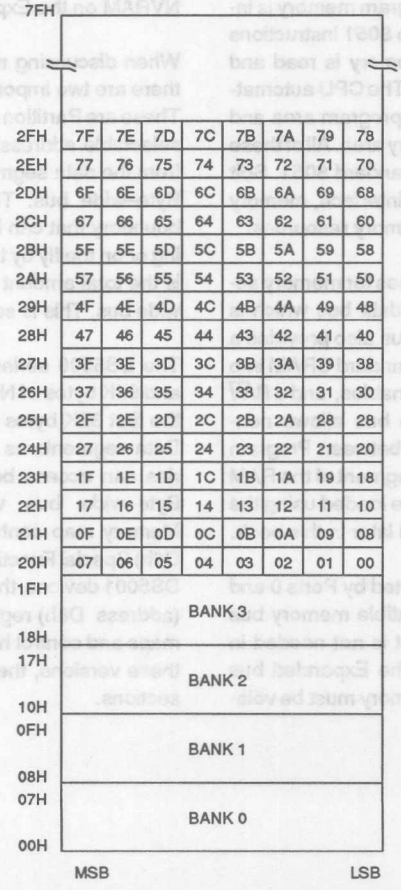
The internal register space is divided into two parts. These are Scratchpad Registers and Special Function Registers (SFRs). There are a total of 128 Scratchpad registers, commonly referred to as on–chip RAM. The 128 bytes include four 8–byte banks of working registers (R0–R7). The Scratchpad Registers are located at register addresses 00–7Fh. This area is not located in the Program or Data Memory area and is accessed by different instructions. The Special Function Registers (SFR) are located in the locations between 80h and FFh. SFRs control the on–chip peripherals and memory configurations. Direct addressing should be used to access the SFR locations. If Register–Indirect addressing is used, indeterminate data will be returned. Scratchpad Registers are discussed immediately below, with SFR descriptions following later in this section.

The Scratchpad Registers are general purpose data storage RAM. They are commonly used for temporary storage of a small number of variables when high–speed access is needed. Off–chip RAM (MOVX) is used when the quantity of data is larger than can be stored in 128 bytes. The Soft Micro will lithium back the Scratchpad area.

The Scratchpad area has two additional functions. First, 16 bytes of the Scratchpad area are bit addressable. That is, while each byte has an address of its own, these bits also have individual bit addresses. Certain micro instructions operate on bits instead of bytes. Although the addresses appear the same, the micro can distinguish a bit address from a byte address by the instruction used. A large number of individual software flags and conditions can be represented using 128 (16*8) individually addressable bits.

A second use of the Scratchpad area is for the programmer's stack. Like the 8051, the Soft Micro uses a Stack Pointer (SP – 81h) SFR to direct stack access into the internal registers. The SP has a default value of 07h. This means that stack storage will begin at location 08h. Each PUSH or CALL instruction will increment the SP. Note that while the SP is located in the SFR area, the stack itself is stored in the Scratchpad area. The Scratchpad Register Memory map is shown in Figure 4–2. *Programmer's note* : with the use of 'C' compilers becoming more frequent, the large memory model should be examined. This compiler model places the stack in off–chip SRAM. Soft Micro based systems usually have an abundance of such SRAM compared to ROM based systems. While off–chip stack results in slower execution time, the stack size becomes virtually unlimited.

SCRATCHPAD REGISTER MAP Figure 4–2



The 8051 instruction set allows efficient (single cycle) access to variables when using the Working Registers. These are a group of four 8–byte banks of Scratchpad RAM. The active Working Registers are referred to as R0–R7. They reside between location 00h and 1Fh, depending on which bank is currently selected. Two bits in the Special Function Register PSW called R1 (PSW.4) and R0 (PSW.3) are used to determine which is the active bank. Once selected, all instructions involving R0–R7 will be directed to the selected group of 8 bytes. This scheme also allows for a fast context switch by simply changing banks. The following Table shows the operation of the Register Bank selection.

PSW.4–3 ; R1–R0

Register Bank Select — Used to select an 8–byte bank of registers to be assigned as R0–R7.

R1	R0	BANK STARTING ADDRESS (R0)
0	0	00h
0	1	08h
1	0	10h
1	1	18h

Program and Data Memory

The Soft Micro divides its main memory between Program and Data segments. Each map consists of a 64K byte area from 0000h to FFFFh. Program memory is inherently read only, since there are no 8051 instructions that write to this segment. Data memory is read and write accessible without restrictions. The CPU automatically routes program fetches to the program area and MOVX instructions to the data memory area. All of these elements are in common with the standard 8051. Soft Micro differences lie in the memory interface, memory map control, and flexibility of the memory resources.

Soft Micros provide two separate buses for memory access. First is a Byte–wide address/data bus which is new to the 8051 architecture. This bus also provides a switched supply output that make standard SRAM into nonvolatile memory, decoded chip enables, and a R/$\overline{W}$ strobe. Furthermore, the Byte–wide bus allows nonvolatile RAM memory to be divided between Program and Data segments. When using a segment of the RAM as Program Memory, this area can be loaded using the Bootstrap Loader function described later in this book.

Second is an Expanded bus constituted by Ports 0 and 2. This is the standard 8051 compatible memory bus which is available as an option, but is not needed in most cases. Program memory on the Expanded bus must be ROM/EPROM and data memory must be volatile SRAM. If NVRAM is needed on the Expanded bus, then it must be externally backed up and write protected. The Soft Micro makes no special provisions for NVRAM on the Expanded bus.

When discussing memory addressing of Soft Micros, there are two important terms that are used frequently. These are Partition and Range. The Partition is the user selectable address that divides the program segment from the data segment in a common RAM area on the Byte–wide bus. The Partition is a user adjustable boundary that can be selected during Bootstrap Loading or on the fly by the application software. The Range is the total amount of memory connected to the Byte–wide bus. This is set once during initial programming.

The DS5000 series devices can access between 8K and 64K bytes of NVRAM on the Byte–wide bus. Up to the first 32K bytes are Partitionable into Program and Data segments as described above. The DS5001 series can access between 8K and 128K bytes on its Byte–wide bus with better Partition control. The Memory map control resides in the MCON (address C6h) Special Function Register on DS5000 devices. On DS5001 devices, the MCON (address C6h) and RPCTL (address D8h) registers are used. Since the memory maps and control have significant differences between these versions, they are described below in separate sections.

DS5000 Series Memory Organization

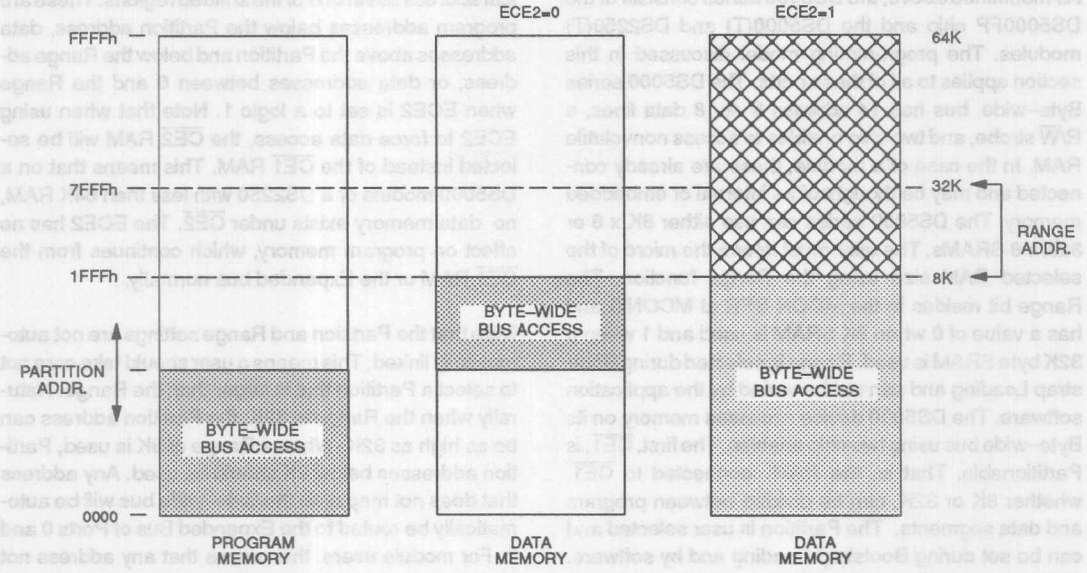
As mentioned above, the DS5000 series consists of the DS5000FP chip and the DS5000(T) and DS2250(T) modules. The programming model discussed in this section applies to all of these parts. The DS5000 series Byte–wide bus has 14 address lines, 8 data lines, a R/$\overline{W}$ strobe, and two chip enables to access nonvolatile RAM. In the case of a module, these are already connected and may be thought of as internal or embedded memory. The DS5000 series can use either 8K x 8 or 32K x 8 SRAMs. The user must inform the micro of the selected RAM size using the Range function. The Range bit resides in the MCON SFR at MCON.3 and has a value of 0 when 8K SRAM is used and 1 when a 32K byte SRAM is used. Range is selected during Bootstrap Loading and can not be varied by the application software. The DS5000 device accesses memory on its Byte–wide bus using two chip enables. The first, $\overline{CE1}$, is Partitionable. That is, the RAM connected to $\overline{CE1}$, whether 8K or 32K, can be divided between program and data segments. The Partition is user selected and can be set during Bootstrap Loading and by software. Partitions are generally available on 2K byte boundaries in the DS5000 except for the last which is 4K. The Partition is selected using the MCON SFR described below. $\overline{CE2}$ is restricted to data memory only. The RAM on $\overline{CE2}$ should be of the same size as $\overline{CE1}$. Access to $\overline{CE2}$ is manual, and functions like a bank switch. Bit 2 (ECE2) of the MCON SFR controls access to $\overline{CE2}$ and is described below.

Figure 4–3 illustrates the functional memory map of a DS5000 series device. The Partition, Range, ECE2, and the logical address combine to determine whether the DS5000 uses its Byte–wide bus or the Expanded Bus. Nonvolatile RAM access will occur when the logical address lies in one of the shaded regions. These are program addresses below the Partition address, data addresses above the Partition and below the Range address, or data addresses between 0 and the Range when ECE2 is set to a logic 1. Note that when using ECE2 to force data access, the $\overline{CE2}$ RAM will be selected instead of the $\overline{CE1}$ RAM. This means that on a DS5000 module or a DS2250 with less than 64K RAM, no data memory exists under $\overline{CE2}$. The ECE2 has no affect on program memory, which continues from the $\overline{CE1}$ RAM or the Expanded bus normally.

Note that the Partition and Range settings are not automatically linked. This means a user should take care not to select a Partition that is larger than the Range. Naturally when the Range is 32K, the Partition address can be as high as 32K. When a Range of 8K is used, Partition addresses below 8K should be used. Any address that does not map onto the Byte–wide bus will be automatically be routed to the Expanded Bus of Ports 0 and 2. For module users. this means that any address not routed to internal memory will got to the ports. The following examples help illustrate the decoding.

When the Partition is at 3000h, and the Range at 32K, program memory below 3000h is accessed on the Byte–wide bus. Program memory at or above 3000h is directed to the Expanded bus or Ports 0 and 2. When the Partition is at 5800h and the Range at 32K, data memory at 0000h is accessed on Ports 0 and 2. Data memory at 6000h is located in NVRAM on the Byte–wide bus. When the Partition is at 1000h and the Range at 8K, all memory access above 1FFFh is on the Expanded bus. Below 8K, the Partition rules apply.

DS5000 SERIES MEMORY MAP Figure 4–3



The above memory map covers the standard operating case. There are two conditions that can modify this memory map. The first is the $\overline{EA}$ pin. The second is the Security Lock. When the $\overline{EA}$ pin is grounded, the DS5000 will force all memory access to the Expanded bus. This causes the DS5000 to behave like an 8031 regardless of the Partition, Range, or ECE2. The $\overline{EA}$ should be pulled to +5V for normal operation. The second modifier is the Security Lock. When set, the Security Lock prevents using the Bootstrap Loader to read the contents of the NVRAM. For security purposes, it also prohibits program memory access on the Expanded Bus. Thus all program fetches must be restricted to the Byte–wide bus when locked. The Security Lock overrides the condition of the $\overline{EA}$ pin as well.

The selection of memory map controls provide unprecedented flexibility to configure a system. However, it is possible to select contradictory settings. The micro will compensate for these as follows. The Partitioning function allows a user to select the quantity of program and data memory. It is possible to select all data and no program in NVRAM by choosing a Partition of 0000h. This is a valid selection. However, using this setting and the Security Lock is a conflict. This condition asks the micro to use all program memory on the Expanded bus, but also to prohibit the use of program memory on the Expanded bus. In this event, special circuits will automatically force the Partition to a location of 7FFFh. This means all 32K memory on the Byte–wide bus is designated program memory. The second contradictory

case is to select a Range of 8K, and to choose a Partition of greater than 8K. This will result in the Range as the limiting factor. Addresses above the Range will automatically be deflected to the Expanded bus. No data memory will be allocated in NVRAM for this configuration.

DS5000 Memory Map Control

The Partition and Range can be selected using the Bootstrap Loader discussed in a later section. In addition, the Partition can be selected or modified by the application software and $\overline{CE2}$ is normally software controlled. However, in either case, the MCON SFR is used to choose these settings. The MCON is summarized in the SFR section below, but appears here also.

DS5000 SERIES MCON REGISTER Figure 4–4

Bit Description:

MCON.7–4: **PA3–0**

"Partition Address": Use to select the starting address of Data Memory in Embedded RAM. Program space lies below the Partition address.

Selection:

PA3	PA2	PA1	PA0	Partition Address
0	0	0	0	0000H
0	0	0	1	0800H
0	0	1	0	1000H
0	0	1	1	1800H
0	1	0	0	2000H
0	1	0	1	2800H
0	1	1	0	3000H
0	1	1	1	3800H
1	0	0	0	4000H
1	0	0	1	4800H
1	0	1	0	5000H
1	0	1	1	5800H
1	1	0	0	6000H
1	1	0	1	6800H
1	1	1	0	7000H*
1	1	1	1	8000H*

*A 4 Kbyte increment (not 2 Kbytes) in the Partition Address take place between bit field values 1110B and 1111B.

Initialization: Set to all 1's on a No V_{LI} Power On Reset or when the Security Lock bit is cleared to a 0 from a previous 1 state. These bits are also set to all 1's when any attempt is made to have them cleared to all 0's with the SL bit set to a 1 (illegal condition).

Read Access: May be read anytime.

Write Access: PAA bit must = 1 in order to write PA3–0. Timed Access is not required to write to PA3–0 once PAA = 1.

MCON.3:	**RA32/8**
"Range Address":	Sets the maximum usable address on the Byte–wide bus. RA32/8 = 0 sets Range Address = 1FFFH (8K); RA32/8 = 1 sets Range Address = 7FFFH (32K)
Initialization:	Set to a 1 on a No V_{LI} Power On Reset and when the Security Lock bit (SL) is cleared to a 0 from a previous 1 state. Remains unchanged on all other types of resets.
Read Access:	May be read normally anytime.
Write Access:	Cannot be modified by the application software; can only be written during Program Load mode.
MCON.2:	**ECE2**
"Enable Chip Enable 2":	Used to enable or disable the $\overline{CE2}$ signal to additional RAM Data Memory space. This bit should always be cleared to 0 in the DS5000–8, DS5000–32, DS2250–8, and DS2250–32 versions.
Initialization:	Cleared to 0 only during a No V_{LI} Power On Reset.
Read Access:	Read normally anytime.
Write Access:	Can be written normally anytime.
MCON.1:	**PAA**
"Partition Address Access":	Used to protect the programming of the Partition Address select bits. PA3–0 cannot be written when PAA=0. PAA can be written only via the Timed Access register.
Initialization:	PAA is cleared on a reset.
Read Access:	PAA may be read anytime.
Write Access:	The Timed Access register must be used to perform any type of write operation on the PAA bit.

DS5001 Series Memory Organization

As mentioned above, the DS5001 series consists of the DS5001FP chip, the DS2251(T) module, the DS5002FP chip, and the DS2252(T) module. Note that the DS5002FP is a high security version of the DS5001FP, but has the same memory map and I/O. The programming model discussed in this section applies to all of these parts and any reference to the DS5001 applies to all of them. The DS5001 series Byte–wide bus has 15 address lines, 8 data lines, a R/$\overline{W}$ strobe, and a total of eight chip enables to access nonvolatile RAM and peripherals. Chip enables include $\overline{CE1}$ – $\overline{CE4}$ and $\overline{PE1}$ – $\overline{PE4}$. The four chip enables ($\overline{CE1}$–4) are for nonvolatile RAM access. How they are connected depends on the memory mode and the selection of SRAMs. The $\overline{PE}$ signals are generally for memory mapped peripherals, but can be used for more RAM if desired. $\overline{PE1}$ and $\overline{PE2}$ are lithium–backed, $\overline{PE3}$ and $\overline{PE4}$ are not. In the case of a module, $\overline{PE1}$ may be connected to a real–time clock. Memory map control resides in the MCON (C6h) and RPCTL (D8h) registers. The MCON register has selected differences from its DS5000 counterpart. These are documented below. The RPCTL is not present in the original DS5000. Also, not all of the bits in this register pertain to memory map control. This section describes the relevant bits and the SFR section below documents the entire register.

The DS5001 series can use multiple 8K x 8 or 32K x 8 SRAMs or a single 128K x 8 SRAM. These parts can operate in either a Partitionable (like DS5000) or non–partitionable mode. The mode is selected via the MCON register by the PM (MCON.1) bit. Note, the DS5001 MCON provides different functions than the

DS5000. In a Partitionable mode (PM=0), the DS5001 can use up to 64K x 8 SRAM for program and data on its Byte–wide bus. It can Partition this area into program and data segments on 4K boundaries. The 64K memory space would consist of two 32K x 8 SRAMs. Each is accessed by a separate chip enable ($\overline{CE1}$ and $\overline{CE2}$), but the micro automatically decodes which is needed. While the DS5001 can use between one 8K x 8 SRAM and 4 32K x 8 SRAMs, it does not automatically know which configuration is used. The Range function determines how much total memory is connected to the Byte–wide bus. The user must identify the total RAM size using the Range bits RG1 and RG0. RG1 is located at MCON.3 and RG0 is located at RPCTL.0. These Range bits are selected during the Bootstrap Loading process and can not be modified by the application software. The Table below shows the Range values that can be selected when PM=0 (Partitionable).

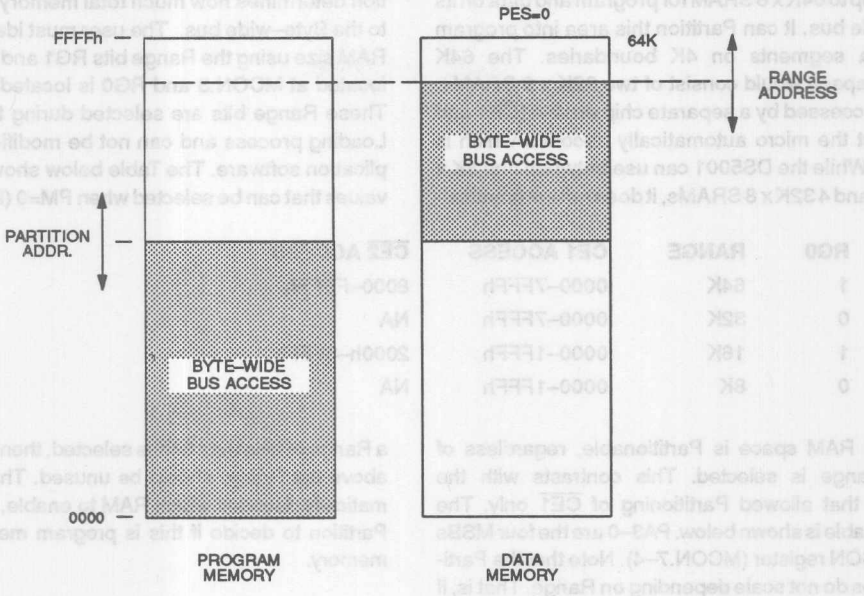
RG1	RG0	RANGE	$\overline{CE1}$ ACCESS	$\overline{CE2}$ ACCESS
1	1	64K	0000–7FFFh	8000–FFFFh
1	0	32K	0000–7FFFh	NA
0	1	16K	0000–1FFFh	2000h–3FFFh
0	0	8K	0000–1FFFh	NA

The total RAM space is Partitionable, regardless of which Range is selected. This contrasts with the DS5000 that allowed Partitioning of $\overline{CE1}$ only. The Partition table is shown below. PA3–0 are the four MSBs of the MCON register (MCON.7–4). Note that the Partition values do not scale depending on Range. That is, if a Range of less than 64K is selected, then the Partitions above the Range should be unused. The micro automatically decodes which RAM to enable, and uses the Partition to decide if this is program memory or data memory.

PA3	PA2	PA1	PA0	PARTITION	BYTE–WIDE BUS MEMORY MAP
0	0	0	0	0000h	0K PROGRAM, DATA = RANGE
0	0	0	1	1000h	4K PROGRAM, DATA = RANGE – 4K
0	0	1	0	2000h	8K PROGRAM, DATA = RANGE – 8K
0	0	1	1	3000h	12K PROGRAM, DATA = RANGE – 12K
0	1	0	0	4000h	16K PROGRAM, DATA = RANGE – 16K
0	1	0	1	5000h	20K PROGRAM, DATA = RANGE – 20K
0	1	1	0	6000h	24K PROGRAM, DATA = RANGE – 24K
0	1	1	1	7000h	28K PROGRAM, DATA = RANGE – 28K
1	0	0	0	8000h	32K PROGRAM, DATA = RANGE – 32K
1	0	0	1	9000h	36K PROGRAM, 28K DATA
1	0	1	0	A000h	40K PROGRAM, 24K DATA
1	0	1	1	B000h	44K PROGRAM, 20K DATA
1	1	0	0	C000h	48K PROGRAM,, 16K DATA
1	1	0	1	D000h	52K PROGRAM, 12K DATA
1	1	1	0	E000h	56K PROGRAM, 8K DATA
1	1	1	1	FFFFh	64K PROGRAM, 0K DATA

Figure 4–5 illustrates the functional memory map of a DS5001 series device in Partitionable mode. Note that like the DS5000, any access that does not correspond to a Byte–wide bus location is routed to the Expanded bus Ports 0 and 2.

PARTITIONABLE MEMORY MAP FOR DS5001 SERIES Figure 4–5

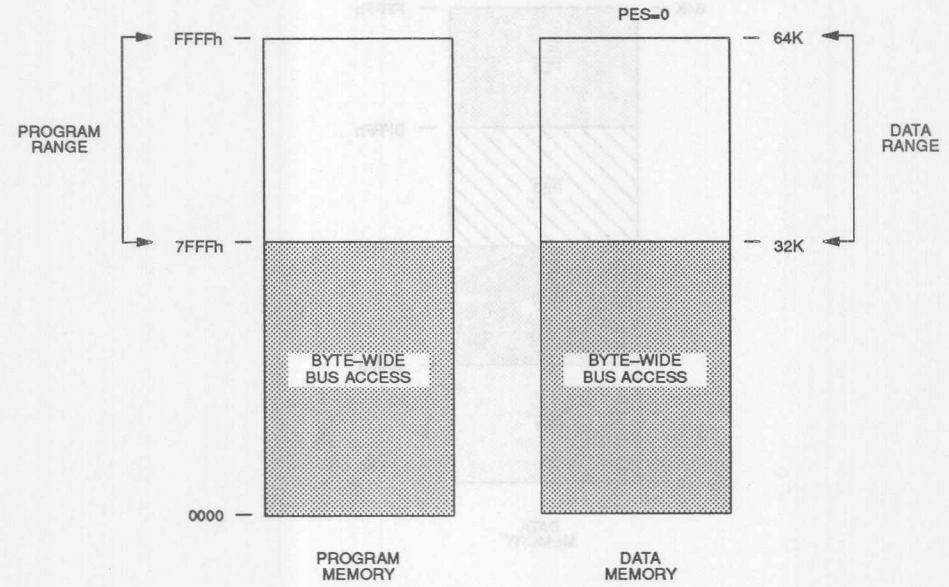


The non–partitionable mode allows the maximum amount of memory to be used on the Byte–wide bus. A non–partitionable mode would be used because the 8051 architecture is restricted to a total of 64K program and 64K data (without bank switching). This means that if the maximum amount of either program or data (or both) is needed, partitioning can not be done. The DS5001 series accommodates these situations with four selections of non–partitionable (PM=1) memory control shown below. These are selected using the Range bits when PM=1. Also note the MSEL signal. This is a pin on DS5001 series devices that tells the processor whether multiple 32K RAMs or a 128K RAM is being used. The four selections are as follows. The non–partitionable memory map is shown in Figure 4–6. Byte–wide bus segments begin at 0000h.

MSEL	RG1	RG0	PROGRAM	DATA	PROGRAM ACCESS	DATA ACCESS
1	0	0	32K	64K	1 @ 32K, $\overline{CE1}$	2 @ 32K, $\overline{CE3}$ and $\overline{CE4}$
1	0	1	64K	32K	2 @ 32K, $\overline{CE1}$ and $\overline{CE2}$	1 @ 32K, $\overline{CE3}$
1	1	0	64K	64K	2 @ 32K, $\overline{CE1}$ and $\overline{CE2}$	2 @ 32K, $\overline{CE3}$ and $\overline{CE4}$
0	1	1	64K	64K	1 @ 128K X 8, for both program and data	

Any address that does not fall into the Byte–wide bus area is routed to the Expanded bus of Ports 0 and 2. This could only occur for the first two settings. Note that the last selection is the most efficient in board space and most likely in cost terms. In this case, all memory addressable by the DS5001 is stored in a nonvolatile 128K x 8 SRAM. In the 128K mode, the DS5001 uses $\overline{CE1}$ as the chip enable, $\overline{CE2}$ is A16, $\overline{CE3}$ is A15, and $\overline{CE4}$ is unused. The micro manipulates these signals automatically when this configuration is selected. This is primarily the reason for the MSEL signal.

NON–PARTITIONABLE MEMORY MAP FOR DS5001 SERIES Figure 4–6

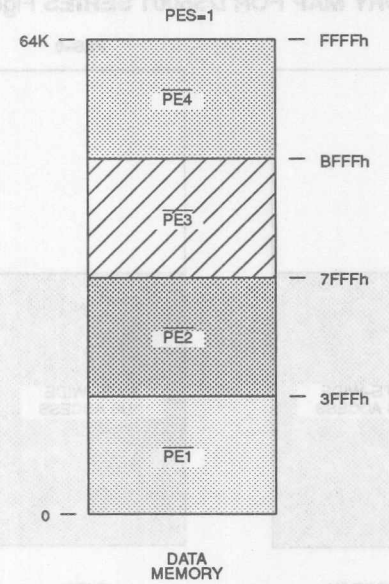


DS5001 Series Memory Mapped Peripherals

The DS5001 series provides four decoded chip enables that can be used for peripheral access or extra RAM on the Byte–wide bus. The four $\overline{PE}$ signals are decoded on 16K boundaries and an internal bank switch is used to access these. A user's application code enables the bank switch to gain access to the Peripheral Selects. While they are enabled, they completely use the data memory map and normal data memory is not available on either the Byte–wide or Expanded bus. The PES bit (MCON.2) is set to a logic 1 to access the peripheral space. When PES=1, the appropriate $\overline{PE}$ signal will be activated based on the logical address. Figure 4–7 shows the data memory map while PES=1. PES has an identical effect for either Partitionable or Non–partitionable modes. It has no effect on the program area. Note that the first two Peripheral Enables, $\overline{PE1}$ and $\overline{PE2}$ are lithium backed by the DS5001. This means that when V_{CC} is removed, the micro will maintain these chip enables in a logic high, inactive state. $\overline{PE3}$ and $\overline{PE4}$ are not lithium backed making them suitable for UARTs, A/Ds, etc. Lithium backed chip enables are used to access lithium backed memory or peripherals. This could include a DS1283 real–time clock which is used in the DS2251T and DS2252T.

On occasion, a memory mapped peripheral is needed that interfaces directly to an 8051 multiplexed bus. When this occurs, MOVX instructions can be forced to use the Expanded bus in any mode with the EXBS bit (RPCTL.5). Setting this bit to a logic one forces all MOVX instructions to the Expanded bus. While EXBS=1, the entire 64K data memory map is accessed in this way. Clearing EXBS will cause the micro to revert to its selected configuration. In most systems, the EXBS bit will not be used.

PERIPHERAL ENABLES IN THE DATA MEMORY MAP Figure 4–7



DS5001 Memory Map Control

Like the DS5000, the DS5001 uses Special Function Registers to control the memory map. The memory control functions include the Partition, Range, Partition Mode (PM), Expanded Bus Select (EXBS), Peripheral Enable Select (PES) and Access Enable (AE – discussed below) . The Partition and Range can be selected using the Bootstrap Loader discussed in a later section. In addition, the Partition can be selected or modified by the application software by writing to the MCON register. PES is normally used by software and is also controlled by the MCON register. The MCON is documented in the SFR summary, but also appears here for convenience. The Range is controlled by a combination of MCON and RPCTL bits. In addition, the EXBS and AE are controlled using the RTPCL register. As not all of the RPCTL bits pertain to memory control, the relevant bits are described below. RPCTL is fully documented in the SFR summary.

DS5001 SERIES MCON REGISTER Figure 4–8

PA3	PA2	PA1	PA0	RG1	PES	PM	—

Bit Description:

MCON.7–4: **PA3–0**
Partition Address. When PM=0, this address specifies the boundary between program and data memory in a continuous space.

Initialization: Unaffected by watchdog, external, or power–up resets. Set to 1111B on a No V_{LI} reset.

Read Access: Can be read normally at any time.

Write Access: Timed Access Protected. Also, cannot be written by the application software if set to 0000B by the serial loader. If a 0000B is written via the serial loader and the security lock is set, the Partition will become 1111B. The same will occur if write access is available and application software writes a 0000B. In addition, these bits will be set to 1111B if security lock is cleared.

MCON.3: **RG1**
One of two bits that determine the range of program space. RG0 is located in the RPCTL register.

Initialization: Unaffected by watchdog, external, or power–up resets. Set to 1 on a No V_{LI} reset or a clearing of the security lock.

Read Access: Can be read at any time.

Write Access: Cannot be modified by the application software. Can only be written during program load.

MCON.2: **PES**
Peripheral Enable Select. When this bit is set, the data space is controlled by $\overline{PE1}$ – $\overline{PE4}$. Peripherals are memory–mapped in 16K blocks, and are accessed by MOVX instructions.

Initialization: Cleared by all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

MCON.1: **PM**
Partition Mode. When PM=0, a partitionable, continuous memory map is invoked. When PM=1, one of four fixed allocations is used.

Initialization: Unaffected by watchdog, external, or power–up reset. Cleared on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by the application software. Can only be modified during program load.

DS5001 SERIES RPCTL REGISTER BITS AFFECTING MEMORY Figure 4–9

RNR	—	EXBS	AE	IBI	DMA	RPCON	RG0

Bit Description:

RPCTL.5: **EXBS**
The Expanded Bus Select routes data memory access (MOVX) to the Expanded bus formed by ports 0 and 2 when set.

Initialization: Cleared after all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

RPCTL.4: **AE**
Access Enable is used when a software reload is desired without using Program Load mode. When set, the DS5001 will be temporarily configured in a Partitionable configuration with the partition at 4K. This will occur even if the PM=1. When cleared, the prior memory configuration is resumed.

Initialization: Cleared after all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time, timed access protected.

RPCTL.0: **RG0**
This is a Range bit which is used to determine the size of the program memory space. Its usage is shown above.

Initialization: Unaffected by watchdog, external, or power–up resets. Cleared on a No V_{LI} reset or clearing of the security lock.

Read Access: Can be read at any time.

Write Access: Cannot be modified by the application software. Can only be written during Program Load.

Loading and Reloading Program Memory

Soft Microcontrollers are programmed through a built–in Bootstrap Loader function. This loader is also used to configure the desired options for memory map control. The Soft Micro uses its low power lithium backed circuits to maintain critical settings in the absence of power. For this reason, it is not necessary to set the Partition, Range, etc. after every power–up or reset. Once set, they will remain unless deliberately modified. Bootstrap Loading is discussed in a later section. One of the major advantages of a Soft Micro is the ability to change these settings, and even reload the entire program memory while the device is installed in system. To completely re–program and re–configure a device, the Bootstrap Loader must be invoked. However, the Soft Micro is designed to allow a partial reload of memory without invoking the Bootstrap Loader.

The major advantage of this technique is that it requires no hardware or external switches. Most of the memory can be reprogrammed under application software control. It would commonly be used when the target system connects to a PC through a serial port as part of an application. For example, a data logger that must dump memory periodically. While connected to the PC, it is extremely easy to reload portions of memory using the "Soft Reload".

Application software always has unrestricted read/write access to the nonvolatile RAM designated as data memory. This is the memory that lies above the Partition address and below the Range address (the non–partitionable configuration of the DS5001 will be addressed separately). Data memory is read or written using the MOVX instruction. Only the area designated as program memory can not be altered. The key to doing a "Soft Reload" of the Soft Micro is to temporarily change the program memory RAM into data memory. Using an SFR, the application software can authorize the Soft Micro to temporarily redefine a portion of the program memory area as data memory. Once this is done, the new code can be received through a serial port (or other means) and written in to data memory. When the process is complete and the new memory is verified as correct, software converts the RAM back into write–protected program memory for the duration. As with the memory map control, there are minor differences between the DS5000 series and DS5001 series devices in how this is accomplished. Each is described below.

SOFT RELOAD OF A DS5000

When application software decides that it should reprogram a portion of memory, the software must convert the target area into data memory. The DS5000 will do this when software sets the PAA bit (MCON.1) to a logic 1. PAA is the Partition Access Enable. Setting PAA has two effects. The micro will automatically move the Partition to 0800h and allow write access to the Partition control bits PA3–0 (MCON.7–4). At this time, the software can adjust the Partition, but the new value will not be used until after PAA is cleared. The Partition remains at 0800h as long as PAA=1, regardless of the Partition control bits. This leaves a 2K block of NVRAM (from 0000–0800h) assigned as program memory. Apart from this, no other changes take place and software continues to operate normally. Caution, make certain that the code that controls the PAA resides in this first 2K. When PAA=1, all addresses on the Byte–wide bus greater than 0800h will be viewed as data memory by the micro and can not be executed even if they were program memory originally. This gives the software read/write access to the remaining 6K bytes (Range=8K) or 30K bytes (Range=32K) of NVRAM on the Byte–wide bus.

At this time, software can begin reloading the target area of memory. There are two minor variations of this procedure. First, a user's loader routine that resides below 0800h (2K) can reprogram the remainder of memory as needed. This is done be receiving the new code through a serial port or other mechanism and writing it to the RAM at the addresses where it will be executed. Since the RAM is data memory, the write operation is done using MOVX instructions.

The second option is that the user's code below 2K can simply move the Partition to a new value. This is done by writing a new value for PA3–0 in MCON (MCON.7–4) while PAA is still set to a 1, then clearing PAA. The purpose of this would be that the loader routine mentioned in option 1 resides in memory above 2K, but below the target memory area. To gain access, the Partition must be moved to a location that includes this loader routine. Once the Partition is moved to this temporary location, the software loader can reprogram new code as before.

When loading is complete, the Partition must be either restored or set to a new value that is appropriate for the new software. If the PA3–0 bits were not modified, then the PAA bit can simply be cleared. This will cause the old Partition to be restored. If the PAA3–0 were modified during loading or software has grown significantly, then a new Partition is needed. The PA3–0 bits must be written while PAA is set to a 1.

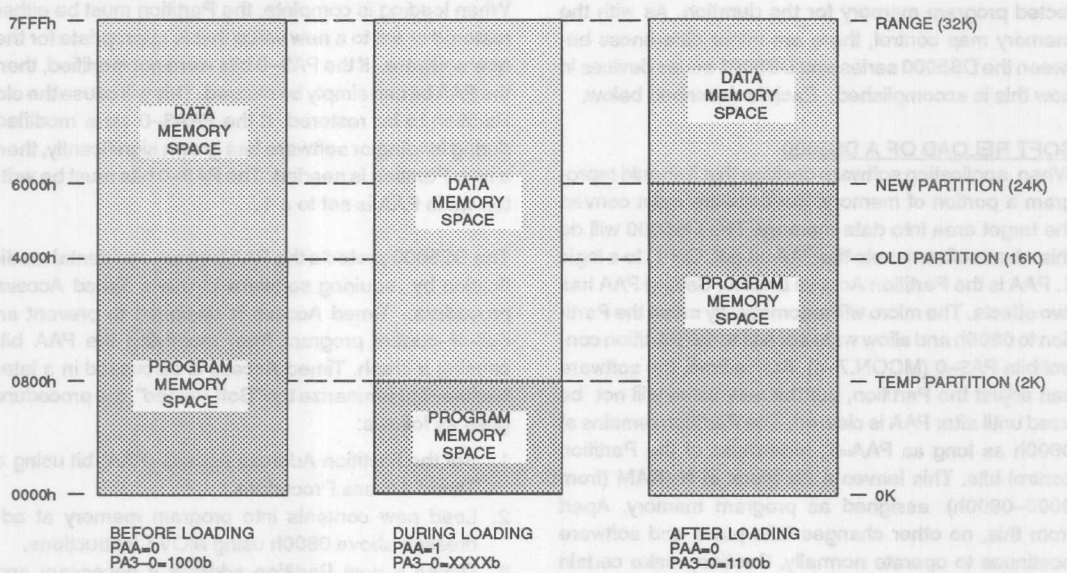
The DS5000 protects the PAA bit from accidental modification by requiring software to use a Timed Access procedure. Timed Access is designed to prevent an out–of–control program from modifying the PAA bit, causing a crash. Timed Access is discussed in a later section. To summarize the "Soft Reload", the procedure goes as follows:

1. Set the Partition Address Access (PAA) bit using a Timed Access Procedure.
2. Load new contents into program memory at addresses above 0800h using MOVX instructions.
3. Define a new Partition address if necessary and write the appropriate bits into PA3–0 in the MCON SFR.
4. Restore the current Partition by clearing the PAA bit with a Timed Access procedure.
5. Resume operation.

The following illustrates an example where a Soft Reload is performed. The original program requires a partition of 4000h (16K bytes). The new program is larger, requiring a Partition of 6000h (24K bytes). The code that performs these steps is shown below. Note that the Timed Access procedure is performed, but is described in a later section.

```
MOV     TA, #0AAh          ; TIMED ACCESS
MOV     TA, #55h           ; TIMED ACCESS 2
MOV     MCON, #10001010b   ; SET PAA BIT
|                          ; USER'S CODE TO LOAD
|                          ; RAM USING MOVX
|
|
MOV     TA, #0AAh          ; TIMED ACCESS
MOV     TA, #55h           ; TIMED ACCESS 2
MOV     MCON, #11001000b   ; LOAD NEW PARTITION AND CLEAR PAA BIT
```

RELOADING PORTIONS OF A DS5000 SERIES DEVICE Figure 4–10



LEGEND:

= NONVOLATILE RAM PROGRAM MEMORY

= NONVOLATILE RAM DATA MEMORY

SOFT RELOAD OF A DS5001

When application software decides that it should reprogram a portion of memory, the software must convert the target area into data memory. However, a Soft Reload of a DS5001 series device has minor variations from the DS5000 version. First, there is no PAA bit in the DS5001. If the DS5001 is in a Partitionable mode then the user's program must manipulate the Partition control bits PA3–0, placing the Partition to a value that permits the target area to be loaded. Moving the Partition to a new value should convert the target area to data memory allowing read/write access. The user's loader routine then uses MOVX instructions to load the new program contents into memory. This program can be received from a serial port or other mechanism. When the loading procedure is complete, a new Partition (or the old one) must be loaded. Note that the loader routine must reside below the Partition at all times.

In the DS5000, the PAA bit was protected by a Timed Access procedure. In the DS5001, the PA3–0 bits are protected directly. The user's program must use a Timed Access procedure to alter these bits. The micro further protects the application by not permitting software to write a 0000b into PA3–0. This would cause a program memory area of 0K. Timed Access is discussed in a later section.

If the DS5001 is in a non–partitionable configuration, then an extra step is required. To perform a Soft Reload of the program contents in a non–partitionable mode, the software must convert the micro to a Partitionable mode temporarily. The Access Enable bit (RPCTL.4) will accomplish this. Setting the AE bit to a logic 1 converts the DS5001 into a Partitionable mode for as long as it is set. This means that regardless of the original setting, once AE=1, the memory map is a 64K partitionable mode. The Partition is set to 1000h (4K) when AE=1, so the loader routine must reside in this area. The user can then perform the Soft Reload as discussed above. When loading is complete, the software should clear the AE bit. Note that AE requires software to use a Timed Access procedure to alter it. This method allows a user to alter program memory in a non–partitionable mode. Data memory can be initialized by application software at any time. Since full read/write access is available, no special provisions are needed.

To summarize the "Soft Reload" for a DS5001, the procedure goes as follows:

Partitionable mode

1. Write a value to PA3–0 using a Timed Access that gives access to the target area of memory.
2. Load new contents into program memory at addresses above the Partition using MOVX instructions.
3. Define a new Partition address if necessary and write the appropriate bits into PA3–0 in the MCON SFR using a Timed Access.
4. Resume operation.

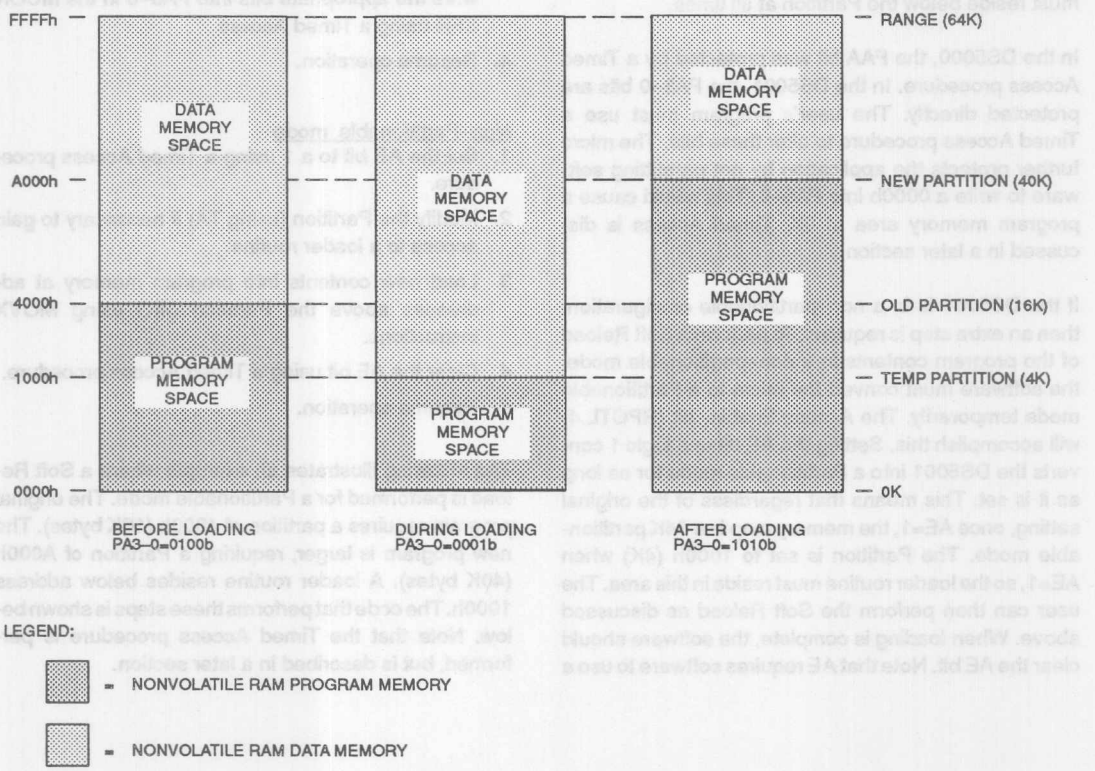
Non–Partitionable mode

1. Set the AE bit to a 1 using a Timed Access procedure.
2. Modify the Partition (using TA) if necessary to gain access to a loader routine.
3. Load new contents into program memory at addresses above the Partition (4K) using MOVX instructions.
4. Clear the AE bit using a Timed Access procedure.
5. Resume operation.

The following illustrates an example where a Soft Reload is performed for a Partitionable mode. The original program requires a partition of 4000h (16K bytes). The new program is larger, requiring a Partition of A000h (40K bytes). A loader routine resides below address 1000h. The code that performs these steps is shown below. Note that the Timed Access procedure is performed, but is described in a later section.

```
MOV     TA, #0AAh           ; TIMED ACCESS
MOV     TA, #55h            ; TIMED ACCESS 2
MOV     MCON, #00011000b    ; SET PARTITION TO 1000h
|                           ; USER'S CODE TO LOAD
|                           ; RAM USING MOVX
|
|
MOV     TA, #0AAh           ; TIMED ACCESS
MOV     TA, #55h            ; TIMED ACCESS 2
MOV     MCON, #10101000b    ; LOAD NEW PARTITION OF A000h
```

RELOADING A DS5001 SERIES DEVICE Figure 4–11



Special Function Registers

The Soft Micro uses Special Function Registers (SFRs) to control most functions. In many cases, an SFR will contain 8 bits, each of which control a function or report status on a function. The SFRs reside in register locations 80–FFh. They can be accessed using MOV instructions with direct addressing. In addition, some of the SFRs are bit addressable. This can be particularly useful when enabling a function without modifying others in the register since an SFR can contain 8 unrelated control and status functions. .

With a few minor exceptions documented below, the Soft Micro provides identical SFRs to a standard 8051, plus extra locations to control unique functions. Modifications to the standard 8051 SFR map are as follows. The PCON register GF1 (PCON.3) and GF0 (PCON.2) have been replaced by the Enable Power Fail Interrupt and the Enable Watchdog Timer bits respectively. In addition, the Soft Micro requires a Timed Access procedure before allowing software to modify the STOP mode bit (PCON.1). This is to prevent errant software from creating a situation that the Watchdog Timer can not recover from. The remaining SFRs are either identical to the 8051 or new to the architecture.

As with the memory map, there are some differences between the DS5000 series and the DS5001 series SFRs. Figures 4–12 and 4–13 show an overview of their respective SFR maps. Following these figures are detailed descriptions. In the case where a particular SFR has differences between the DS5000 and DS5001, those differences will be pointed out under the particular register. In some cases, the DS5001 has registers that do not appear in the DS5000. This is also highlighted under the particular register.

DS5000 SERIES SPECIAL FUNCTION REGISTER MAP Figure 4–12

DIRECT BYTE ADDRESS	(MSB) BIT ADDRESS (LSB)								SPECIAL FUNCTION REGISTER SYMBOL
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	ACC
	C	AC	F0	RS1	RS0	OV		P	
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
0C7H	NOT BIT ADDRESSABLE								TA
	PA3	*PA2*	*PA1*	*PA0*	*RA32/8*	*ECE2*	PAA	*SL*	
0C6H	NOT BIT ADDRESSABLE								MCON
	RWT			PS	PT1	PX1	PT0	PX0	
0B8H	BF	–	–	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
	EA			ES	ET1	EX1	ET0	EX0	
0A8H	AF	–	–	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
99H	NOT BIT ADDRESSABLE								SBUF
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
8DH	NOT BIT ADDRESSABLE								TH1
8CH	NOT BIT ADDRESSABLE								TH0
8BH	NOT BIT ADDRESSABLE								TL1
8AH	NOT BIT ADDRESSABLE								TL0
	GATE	$\overline{C/T}$	M1	M0	GATE	$C/\overline{T}$	M1	M0	
89H	NOT BIT ADDRESSABLE								TMOD
	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
88H	8F	83	8D	8C	8B	8A	89	88	TCON
	SMOD	$\overline{POR}$	PFW	*WTR*	EPFW	*EWT*	STOP	IDL	
87H	NOT BIT ADDRESSABLE								PCON
83H	NOT BIT ADDRESSABLE								DPH
82H	NOT BIT ADDRESSABLE								DPL
81H	NOT BIT ADDRESSABLE								SP
80H	87	86	85	84	83	82	81	80	P0

* BITS IN ITALICS ARE NONVOLATILE

DS5001 SERIES SPECIAL FUNCTION REGISTER MAP Figure 4–13

DIRECT BYTE ADDRESS	BIT ADDRESS (MSB)							(LSB)	SPECIAL FUNCTION REGISTER SYMBOL
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	ACC
	ST7	ST6	ST5	ST4	IA0	F0	IBF	0BF	
0DAH	NOT BIT ADDRESSABLE								STATUS
	RNR	—	EXBS	AE	IBI	DMA	RPC	*RG0*	
0D8H	DF	DE	DD	DC	DB	DA	D9	D8	RPCTL
	C	AC	F0	RS1	RS0	OV		P	
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
0CFH	NOT BIT ADDRESSABLE								RNR
0C7H	NOT BIT ADDRESSABLE								TA
	PA3	*PA2*	*PA1*	*PA0*	*RG1*	PES	*PM*	*SL*	
0C6H	NOT BIT ADDRESSABLE								MCON
0C3H	NOT BIT ADDRESSABLE								CRC HIGH
0C2H	NOT BIT ADDRESSABLE								CRC LOW
	RNGE3	RNGE2	RNGE1	RNGE0	—	—	*MDM*	*CRC*	
0C1H	NOT BIT ADDRESSABLE								CRC
	RWT			PS	PT1	PX1	PT0	PX0	
0B8H	BF	—	—	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
	EA			ES	ET1	EX1	ET0	EX0	
0A8H	AF	—	—	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
99H	NOT BIT ADDRESSABLE								SBUF
	SM0	SM1	SM2	REN	TB8	RB8	TL	RI	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
8DH	NOT BIT ADDRESSABLE								TH1
8CH	NOT BIT ADDRESSABLE								TH0
8BH	NOT BIT ADDRESSABLE								TL1
8AH	NOT BIT ADDRESSABLE								TL0
	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	
89H	NOT BIT ADDRESSABLE								TMOD
	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
	SMOD	$\overline{POR}$	PFW	*WTR*	EPFW	*EWT*	STOP	IDL	
87H	NOT BIT ADDRESSABLE								PCON
83H	NOT BIT ADDRESSABLE								DPH
82H	NOT BIT ADDRESSABLE								DPL
81H	NOT BIT ADDRESSABLE								SP
	D7	D6	D5	D4	D3	D2	D1	D0	
80H	87	86	85	84	83	82	81	80	P0/DBB

* BITS IN ITALICS ARE NONVOLATILE

POWER CONTROL REGISTER

Label: PCON **Register Address: 087H**

D7	D6	D5	D4	D3	D2	D1	D0
SMOD	POR	PFW	WTR	EPFW	EWT	STOP	IDL

Bit Description:

PCON.7 **SMOD**

"Double Baud Rate": When set to a 1, the baud rate is doubled when the serial port is being used in modes 1, 2, or 3.

Initialization: Cleared to a 0 on any reset.

Read Access: Can be read normally at any time.

Write Access: Can be written normally at any time.

PCON.6 **$\overline{\text{POR}}$**

"Power On Reset": Indicates that the previous reset was initiated during a Power On sequence.

Initialization: Cleared to a 0 when Power On Reset occurs. Remains at 0 until it is set to a 1 by software.

Read Access: Can be read normally at any time.

Write Access: Can be written only by using the Timed Access Register.

PCON.5: **PFW**

"Power Fail Warning": Indicates that a potential power failure is in progress. Set to 1 whenever V_{CC} voltage is below the V_{PFW} threshold. Cleared to a 0 immediately following a read operation of the PCON register. Once set, it will remain set until the read operation occurs regardless of activity on V_{CC}. After PFW is cleared by a read, it will return to a 1 if $V_{CC} < V_{PFW}$.

Initialization: Cleared to a 0 during a Power On Reset.

Read Access: Can be read normally anytime.

Write Access: Not writable.

PCON.4: **WTR**

"Watchdog Timer Reset": Set to a 1 following a Watchdog Timer timeout. If Watchdog Timer Reset is enabled, this will indicate the cause of the reset. Cleared to 0 immediately following a read of the PCON register.

Initialization: Set to a 1 after a Watchdog Timeout Reset. Cleared to a 0 on a Power On Reset. Remains unchanged during other types of resets.

Read Access: May be read normally anytime.

Write Access: Cannot by written.

PCON.3:	**EPFW**
"Enable Power Fail Interrupt":	Used to enable or disable the Power Fail Interrupt. When EPFW is set to a 1, it will be enabled; it will be disabled when EPFW is cleared to a 0.
Initialization:	Cleared to a 0 on any type of reset.
Read Access:	Can be read normally anytime.
Write Access:	Can be written normally anytime.
PCON.2:	**EWT**
"Enable Watchdog Timer":	Used to enabled or disable the Watchdog Timeout Reset. The Watchdog Timer is enabled if EWT is set to a 1 and will be disabled if EWT is cleared to a 0.
Initialization:	Cleared to a 0 on a No-V_{LI} Power on Reset. Remains unchanged during other types of reset.
Read Access:	May be read normally anytime.
Write Access:	Can be written only by using the Timed Access register.
PCON.1:	**STOP**
"Stop":	Used to invoke the Stop mode. When set to a 1, program execution will terminate immediately and Stop mode operation will commence. Cleared to a 0 when program execution resumes following a hardware reset.
Initialization:	Cleared to a 0 on any type of reset.
Read Access:	Can be read anytime.
Write Access:	Can be written only by using the Timed Access register.
PCON.0:	**IDL**
"Idle"	Used to invoke the Idle mode. When set to a 1, program execution will be halted and will resume when the idle bit is cleared to 0 following an interrupt or a hardware reset.
Initialization:	Cleared to 0 on any type of reset or interrupt.
Read Access:	Can be read normally anytime.
Write Access:	Can be written normally anytime.

TIMER CONTROL REGISTER

Label: TCON **Register Address 088H**

D7	D6	D5	D4	D3	D2	D1	D0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Bit Description:

TCON.7: **TF1**

"Timer 1 Overflow Flag": Status bit set to 1 when Timer 1 overflows from a previous count value of all 1's. Cleared to 0 when CPU vectors to Timer 1 interrupt service routine.

Initialization: Cleared to 0 on any type of reset.

TCON.6: **TR1**

"Timer 1 Run Control": When set to a 1 by software, Timer 1 operation will be enabled. Timer 1 is disabled when cleared to 0.

Initialization: Cleared to 0 on any type of reset.

TCON.5: **TF0**

"Timer 0 Overflow": Status bit set to 1 when Timer 0 overflows from a previous count value of all 1's. Cleared to 0 when CPU vectors to Timer 0 interrupt service routine.

Initialization: Cleared to 0 on any type of reset.

TCON.4: **TR0**

"Timer 0 Run Control": When set to a 1 by software, Timer 0 operation is enabled. Timer 0 is disabled when cleared to 0.

Initialization: Cleared to 0 on any type of reset.

TCON.3: **IE1**

"Interrupt 1 Edge Detect": Set to 1 to signal when a 1-to-0 transition (IT=1) or a low level (IT=0) has been detected on the $\overline{INT1}$ pin. Cleared to a 0 by hardware when interrupt processed only if IT1=1.

Initialization: Cleared to 0 on any type of reset.

TCON.2: **IT1**

"Interrupt 1 Type Select": When set to 1, 1-to-0 transitions on $\overline{INT1}$ will be used to generate interrupt requests from this pin. When cleared to 0, $\overline{INT1}$ is level-activated.

Initialization: Cleared to a 0 on any type of reset.

TCON.1: **IE0**

"Interrupt 0 Edge Detect": Set to a 1 to signal when a 1-to-0 transition (IT0=1) or a low level (IT0=0) has been detected on the $\overline{INT0}$ pin. Cleared to a 0 by hardware when interrupt processed only if IT0=1.

Initialization: Cleared to a 0 on any type of reset.

TCON.0: **IT0**

"Interrupt 0 Type Select": When set to 1, 1-to-0 transitions on INT0 will be used to generate interrupt requests from this pin. When cleared to 0, $\overline{INT0}$ is level-activated.

Initialization: Cleared to a 0 on any type of reset.

TIMER MODE REGISTER

Label: TMOD **Register Address: 089H**

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0

Bit Description:

TMOD.7 (Timer 1);
TMOD.3 (Timer 0): **GATE**

"Gate Control": When set to 1 with TRn=1, timer/counter's input count pulses will only be delivered while a 1 is present on the $\overline{INT}$ pin. When cleared to 0, count pulses will always be received by the timer/counter a long as TRn=1..

Initialization: Cleared to 0 on any reset.

TMOD.6 (Timer 1);
TMOD.2 (Timer 0) **C/$\overline{T}$**

"Counter/Timer Select": When set to 1, the counter function is selected for the associated programmable timer; when cleared to 0, the timer function is selected.

Initialization: Cleared to 0 on any reset.

TMOD.5, TMOD.4 (Timer 1);
TMOD.1, TMOD.0 (Timer 0): **M1,M0**

"Mode Select": These bits select the operating mode of the associated timer/counter as follows:

M1	M0	
0	0	Mode 0: 8 bits with 5-bit prescale
0	1	Mode 1: 16 bits with no prescale
1	0	Mode 2: 8 bits with auto-reload
1	1	Mode 3: Timer 0 – Two 8-bit timers Timer 1 – Stopped

Initialization: Cleared to 0 on any reset.

SERIAL CONTROL REGISTER

Label: SCON **Register Address: 098H**

D7	D6	D5	D4	D3	D2	D1	D0
SM0	SM	SM2	REN	TB8	RB8	TI	RI

Bit Description:

SCON.7, SCON.6: **SM0, SM1**

"Mode Select": Used to select the operational mode of the serial I/O port as follows:

SM0	SM1	MODE	WORD FUNCTION	BAUD LENGTH	CLOCK PERIOD
0	0	Mode 0	SYNC	8 bits	12 t_{CLK}
0	1	Mode 1	ASYNC	10 bits	Timer 1 Overflow
1	0	Mode 2	ASYNC	11 bits	64 t_{CLK} or 32 t_{CLK}
1	1	Mode 3	ASYNC	11 bits	Timer 1 Overflow

Initialization: Cleared to 0 on any type of reset.

SCON.5: **SM2**

"Multiple MCU Comm": Used to enable the multiple microcontroller communications feature for modes 2 and 3. When SM2=1, RI will be activated only when serial words are received which cause RB8 to be set to a 1.

Initialization: Cleared to a 0 on any type of reset.

SCON.4: **REN**

"Receive Enable": When set to 1, the receive shift register will be enabled. Disabled when cleared to 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.3: **TB8**

"Xmit Bit 8": Can be set or cleared to define the state of the 9th data bit in modes 2 and 3 of a serial data word.

Initialization: Cleared to a 0 on any type of reset.

SCON.2: **RB8**

"Rec. Bit 8": Indicates the state of the 9th data bit received while in modes, 2 or 3. If mode 1 is selected with SM2=0, RB8 is the state of the stop bit which was received. RB8 is not used in mode 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.1: **TI**

"Xmit Interrupt": Status bit used to signal that a data word has been completely shifted out. In mode 0, it is set at the end of the 8th data bit. Set when the stop bit is transmitted in all other modes.

Initialization: Cleared to a 0 on any type of reset.

SCON.0: **RI**

"Receive Interrupt": Status bit used to signal that a serial data word has been received and loaded into the receive buffer register. In mode 0, it is set at the end of the 8th bit time. It is set at the mid-bit time of the incoming stop bit in all other modes of a valid received word according to the state of SM2.

INTERRUPT ENABLE REGISTER

Label: IE **Register Address: 0A8H**

D7	D6	D5	D4	D3	D2	D1	D0
EA	–	–	ES	ET1	EX1	ET0	EX0

Bit Description:

IE.7: **EA**

"Enable All Interrupts": When set to 1, each interrupt except for PFW may be individually enabled or disabled by setting or clearing the associated IE.x bit. When cleared to 0, interrupts are globally disabled and no pending interrupt request will be acknowledged except for PFW.

IE.4: **ES**

"Enable Serial Interrupt": When set to 1, an interrupt request from either the serial port's TI or RI flags can be acknowledged. Serial I/O interrupts are disabled when cleared to 0.

IE.3: **ET1**

"Enable Timer 1 Interrupt": When set to 1, an interrupt request from Timer 1's TF1 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.2: **EX1**

"Enable External Interrupt 1": When set to 1, an interrupt request from the IE1 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.1: **ET0**

"Enable Timer 0 Interrupt": When set to 1, an interrupt request from Timer 0's TF0 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.0: **EX0**

"Enable External Interrupt 0": When set to 1, an interrupt from the IE0 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

INTERRUPT PRIORITY REGISTER

Label: IP **Register Address: 0B8H**

D7	D6	D5	D4	D3	D2	D1	D0
RWT	–	–	PS	PT1	PX1	PT0	PX0

Bit Description:

IP.7: **RWT**

"Reset Watchdog Timer": When set to a 1, the Watchdog Timer count will be reset and counting will begin again. The RWT bit will then automatically be cleared again to 0. Writing a 0 into this bit has no effect.

Initialization: Cleared to a 0 on any reset.

Read Access: Cannot be read.

Write Access: Can be written only by using the Timed Access register.

All of the following bits are read/write at any time and are cleared to 0 following any hardware reset.

IP.4: **PS**

"Serial Port Priority": Programs Serial Port interrupts for high priority when set to 1. Low priority is selected when cleared to 0.

IP.3: **PT1**

"Timer 1 Priority": Programs Timer 1 interrupt for high priority when set to 1. Low priority is selected when cleared to 0.

IP.2: **PX1**

"Ext. Int. 1 Priority": Programs External Interrupt 1 for high priority when set to 1. Low priority is selected when cleared to 0.

IP.1: **PT0**

"Timer 0 Priority": Programs Timer 0 Interrupt for high priority when set to 1. Low priority is selected when cleared to 0.

IP.0: **PX0**

"Ext. Int. 0 Priority" Programs External Interrupt 0 for high priority when set to 1. Low priority is selected when cleared to 0.

DS5001 CRC REGISTER

Label: CRC **Register Address: 0C1H**

RNGE3	RNGE2	RNGE1	RNGE0	—	—	MDM	CRC

Bit Description:

CRC.7–4 **RNGE3–0**
Determines the range over which a power–up CRC will be performed. Addresses are specified on 4K boundaries.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by the application software. Can only be written via the Bootstrap Loader.

CRC.1 **MDM**
When set to 1, the bootstrap loader will attempt to use a modem (UART) on $\overline{PE4}$ if CRC is incorrect.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by the application software. Can only be written during Program Load mode.

CRC.0 **CRC**
When set to 1, a CRC check will be performed on power–up or watchdog timeout. CRC will be checked to stored values. An error will initiate Program Load mode.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by the application software. Can only be written during Program Load mode.

DS5000 MEMORY CONTROL REGISTER

Label: MCON **Register Address: 0C6H**

D7	D6	D5	D4	D3	D2	D1	D0
PA3	PA2	PA1	PA0	RA32/8	ECE2	PAA	SL

Bit Description:

MCON.7–4: **PA3–0**

"Partition Address": Used to select the starting address of Data Memory on the Byte–wide bus. Program space lies below the partition address.

PA3	PA2	PA1	PA0	Partition Address
0	0	0	0	0000H
0	0	0	1	0800H
0	0	1	0	1000H
0	0	1	1	1800H
0	1	0	0	2000H
0	1	0	1	2800H
0	1	1	0	3000H
0	1	1	1	3800H
1	0	0	0	4000H
1	0	0	1	4800H
1	0	1	0	5000H
1	0	1	1	5800H
1	1	0	0	6000H
1	1	0	1	6800H
1	1	1	0	7000H*
1	1	1	1	8000H*

*A 4 Kbyte increment (not 2 Kbytes) in the Partition Address takes place between bit field values 1110B and 111B.

Initialization: Set to all 1's on a No V_{LI} Power On Reset or when the Security Lock bit is cleared to a 0 from previous 1 state. These bits are also set to all 1's when any attempt is made to have them cleared to all 0's with the SL bit set to 1 (illegal condition).

Read Access: May be read anytime.

Write Access: PAA bit must = 1 in order to write PA3–0. Timed Access is not required to write to PA3–0 once PAA=1.

MCON.3: **RA32/8**

"Range Address": Set the maximum usable address in on the Byte–wide bus. RA32/8=0 sets Range Address = 1FFFH (8K)
RA32/8 = 1 sets Range Address = 7FFFH (32K)

Initialization: Set to a 1 during a No V_{LI} Power On Reset and when the Security Lock bit (SL) is cleared to a 0 from a previous 1 state. Remains unchanged on all other types of resets.

Read Access: May be read normally anytime.

Write Access: Cannot be modified by the application software; can only be written via the Bootstrap Loader.

MCON.2: **ECE2**

"Enable Chip Enable 2": Used to enable or disable the $\overline{CE2}$ signal for the Byte–wide bus data memory. This bit should always be cleared to 0 in the DS5000, DS5000-32, DS2250–8 and DS2250–32 versions.

Initialization: Cleared to 0 only during a No V_{LI} Power On Reset.

Read Access: Read normally anytime.

Write Access: Can be written normally at any time.

MCON.1: **PAA**

"Partition Address Access": Used to protect the programming of the Partition Address select bits. PA3–0 cannot be written when PAA=0. PAA can be written only via the Timed Access register.

Initialization: PAA is cleared on any reset.

Read Access: PAA may be read anytime.

Write Access: The Timed Access register must be used to perform any type of write operation on the PAA bit.

MCON.0: **SL**

"Security Lock": Indicates that the security lock is set when SL=1.

Initialization: Cleared to a 0 on a no V_{LI} power on reset.

Read Access: Read normally any time.

Write Access: Can only be written by the Bootstrap loader.

DS5001 MCON REGISTER

Label: MCON **Register Address: 0C6H**

PA3	PA2	PA1	PA0	RG1	PES	PM	SL

Bit Description:

MCON.7–4: **PA3–0**
Partition Address. When PM=0, this address specifies the boundary between program and data memory in a continuous space.

Initialization: Unaffected by watchdog, external, or power–up resets. Set to 1111B on a No V_{LI} reset.

Read Access: Can be read normally at any time.

Write Access: Timed Access Protected. Cannot be written by the application software if set to 0000B by the serial loader. If a 0000B is written via the serial loader and the security lock is set, the Partition will become 1111B. The same will

occur if write access is available and application software writes a 0000B. In addition, these bits will be set to 1111B if security lock is cleared.

MCON.3: **RG1**
One of two bits that determine the range of program space. RG0 is located in the RPCTL register.

Initialization: Unaffected by watchdog, external, or power–up resets. Set to 1 on a No V_{LI} reset or a clearing of the security lock.

Read Access: Can be read at any time.

Write Access: Cannot be modified by the application software. Can only be written via the Bootstrap Loader.

MCON.2: **PES**
Peripheral Enable Select. When this bit is set, the data space is controlled by $\overline{PE1}$–$\overline{PE4}$. Peripherals are memory–mapped in 16K blocks, and are accessed by MOVX instructions on the Byte–wide bus..

Initialization: Cleared by all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

MCON.1: **PM**
Partition Mode. When PM=0, a partitionable, continuous memory map is invoked. When PM=1, one of four fixed allocations is used.

Initialization: Unaffected by watchdog, external, or power–up resets. Cleared on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be modified by the application software. Can only be modified via the Boostrap Loader.

MCON.0: **SL**

"Security Lock": Indicates that the security lock is set when SL=1.

Initialization: Cleared to a 0 on a no V_{LI} power on reset.

Read Access: Read normally any time.

Write Access: Can only be written by the Bootstrap loader.

PROGRAM STATUS WORD REGISTER

Label: PSW **Register Address: 0D0H**

D7	D6	D5	D4	D3	D2	D1	D0
C	AC	F0	RS1	RS0	OV		P

All of the bits in PSW are read/write and are cleared to 0 on any type of reset.

Bit Description:

PSW.7: **C**

"Carry": Set when the previous operation resulted in a carry (during addition) or a borrow (during subtraction). Otherwise cleared.

PSW.6: **AC**

"Auxiliary-Carry": Set when the previous operation resulted in a carry (during addition) or a borrow (during subtraction) from the high-order nibble. Otherwise cleared.

PSW.5: **F0**

"User Flag 0": General-purpose flag bit which can be set or cleared as needed.

PSW.4–3: **R1–R0**

"Register Bank Select": Used to select an 8-byte bank of registers within the Data Register space to be assigned as R0–R8 in subsequent instructions. The 8-byte bank starting address selection is as follows:

R1	R0	Data Register Address (R0)
0	0	00H
0	0	08H
1	0	10H
1	1	18H

PSW.2: **OV**

"Overflow": Set when a carry was generated into the high-order bit but not a carry out of the high-order bit as a result of the previous operation, and visa-versa. OV is normally used in 2's complement arithmetic.

PSW.0: **P**

"Parity": Set if the modulo-2 sum of the eight bits of the accumulator is 1 (odd parity); cleared on even parity.

DS5001 RPC CONTROL REGISTER

Label: RPCTL **Register Address: 0D8H**

RNR	—	EXBS	AE	IBI	DMA	RPCON	RGO

Bit Description:

RPCTL.7 **RNR**
The random number generator of the DS5001FP is available to the user. When a random number is required, the RNR bit signifies that one is available. This bit is cleared when the random number is read, and approximately 160 µsec are required to generate the next number.

Initialization: Cleared after all resets. Bit will be set approximately 160 µsec after a reset.

Read Access: Can be read at any time.

Write Access: Cannot be written.

RPCTL.5 **EXBS**
The Expanded Bus Select routes data memory access (MOVX) to the expanded bus formed by ports 0 and 2 when set.

Initialization: Cleared after all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

RPCTL.4 **AE**
Access Enable is used when a software reload is desired without using the Bootstrap Loader. When set, the DS5001 will be temporarily configured in a Partitionable configuration with the Partition at 4K. This will occur even if the PM=1. When cleared, the prior memory configuration is resumed.

Initialization: Cleared after all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time, Timed Access protected.

RPCTL.3 **IBI**
When using the RPC mode, an interrupt may be required for the Input Buffer Flag. This interrupt is enabled by setting the Input Buffer Interrupt (IBI) bit. At this time, the timer 1 interrupt is disabled, and this RPC mode interrupt is used in its place (vector location 1BH). This bit can be set only when the RPCON bit is set.

Initialization: Cleared on all resets, and when the RPCON bit is cleared.

Read Access: Can be read at any time.

Write Access: Can be written when the RPC mode is enabled (RPCON=1).

RPCTL.2 **DMA**
This bit is set to enable DMA transfers when RPC mode is invoked. It can only be set when RPCON=1.

Initialization: Cleared on all resets, and when RPC is cleared.

Read Access: Can be read anytime.

Write Access: Can be written when the RPC mode is enabled (RPCON=1).

RPCTL.1 **RPCON**
Enable the RPC 8042 I/O protocol. When set, port 0 becomes the data bus, and port 2 becomes the control signals.

Initialization: Cleared on all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

RPCTL.0 **RG0**
This is a Range bit which is used to determine the size of the program memory space. Its usage is shown above.

Initialization: Unaffected by watchdog, external, or power–up resets. Cleared on a No V_{LI} reset or clearing of the security lock.

Read Access: Can be read at any time.

Write Access: Cannot be modified by the application software. Can only be written during Program Load.

DS5001 RPC STATUS REGISTER

Label: RPS **Register Address: 0DAH**

ST7	ST6	ST5	ST4	IA0	F0	IBF	OBF

Bit Description:

RPS.7–4: General purpose status bits that can be written by the DS5001FP and can be read by the external host.

Initialization: Cleared when RPCON=0.

Read Access: Can be read by DS5001 and host CPU when RPC mode is invoked.

Write Access: Can be written by the DS5001 when RPC mode is invoked.

RPS.3: **IA0**
Stores the value of the external system A0 for the last DBBIN Write when a valid write occurs (as determined by the IBF flag).

Initialization: Cleared when RPC=0.

Read Access: Can be read by DS5001 and host CPU when in RPC mode.

Write Access: Automatically written when a valid DBBIN Write occurs. Cannot be written otherwise.

RPS.2: **F0**
General purpose flag written by the DS5001 and read by the external host.

Initialization: Cleared when RPC=0.

Read Access: Can be read by DS5001 and host CPU when in RPC mode.

Write Access:	Can be written by the DS5001 when in RPC mode.
RPS.1:	**IBF** Input Buffer Full Flag is set following a write by the external host, and is cleared following a read of the DBBIN by the DS5001.
Initialization:	Cleared when RPC=0.
Read Access:	Can be read by DS5001 and host CPU when in RPC mode.
Write Access:	Written automatically as part of the RPC communication. Cannot be set by the application software.
RPS.0:	**OBF** Output Buffer Full Flag is set following a write of the DBBOUT by the DS5001, and is cleared following a read of the DBBOUT by the external host.
Initialization:	Cleared when RPC=0.
Read Access:	Can be read by DS5001 and host CPU when in RPC mode.
Write Access:	Written automatically as part of the RPC communication. Cannot be set by the application software.

INSTRUCTION SET

Introduction

The Soft Micro executes an instruction set which is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages which have been written for the 8051 are compatible with the Soft Micro including cross-assemblers, high-level language compilers, and debugging tools.

There are a total of 42 instruction types recognized by the Soft Micro. When the instruction uses both source and destination operands, they are specified in the order of "destination, source".

Addressing Modes

There are eight addressing modes implemented in the instruction set executed by the Soft Micro. Five of these are used to address operands. The other three are used in instructions which transfer execution of the program to another address (e.g., Branch, Jump, Call).

The modes which address source operands, include Register Addressing, Direct Addressing, Register-Indirect Addressing, Immediate Addressing and Register-Indirect with Displacement. The first three of these can also be used to address a destination operand. Most instructions use operands that are located in the Internal Data Registers.

The addressing modes used for the Control Transfer instructions include Relative Addressing, Page Addressing, and Extended Addressing.

The operation of these addressing modes is summarized below.

Register Addressing

Register Addressing is used on operands contained in one of the eight registers (R7–R0) of the currently selected Working Register Bank. A register bank is selected by programming a 2-bit field in the PSW Special Function register. All of the Working registers may also be accessed through either Direct Addressing or Register-Indirect Addressing as well. This is due to the fact that the Working registers are mapped into the lower 32 bytes of Internal Data RAM as discussed above. An example of an instruction which employs register addressing is:

```
ADD   A,R4      ; Add Accumulator to Working
                ; register R4
```

Direct Addressing

Direct Addressing is the only mode available for use on operands within the Special Function registers. Addressing of bytes may also be used to access the 128 Internal Data registers. An example of an instruction which utilizes Direct Register Addressing is:

```
MOV   072H,074H ; Load direct register (addr. 072H)
                ; with direct register (074H)
```

Direct addressing of bits is available on 128 bits located in the Internal Data registers in byte addresses of 20H – 2FH inclusive. Direct bit addressing is also available in Special Function registers located at addresses on 8-byte boundaries starting at 80H (i.e., 80H, 88H, 90H, 98H, ...0F0H, 0F8H). An examples of an instruction which uses direct bit addressing is:

```
SETB  00H       ; Set addressable bit 00H (D0 in
                ; Internal Data Reg. 20H)
```

Register Indirect Addressing

Some instructions use Register-Indirect Addressing for accessing operands in other Internal Data registers. This is done by using the contents of Working register R1 or R0 as a pointer to other Internal Data registers. An example of this is:

```
ANL   A,@R0     ; Logical AND of Accumulator with
                ; Internal Data register; pointed to
                ; by contents of R0
```

In addition, this addressing is used via the Stack Pointer register (SP) for manipulation of the stack. The stack area is contained in the Internal Data Register area. The PUSH and POP instructions are the only ones which use SP for this addressing mode. As an example:

```
PUSH  P0        ; Save the contents of the Port 0
                ; SFR latch on the stack
```

The R0, R1, and the DPTR registers are used with Register-Indirect Addressing for accessing Data Memory. R1 or R0 in the selected Working Register bank may be used for accessing location within a 256-byte block

pointed to by the current contents of the P2 SFR latch (address high byte). For example:

```
MOVX  A,@R1      ; Load the Accumulator with the
                 ; contents of Data Memory
                 ; addressed by the 8-bit contents
                 ; of R1
```

The 16-bit DPTR register may be used to access any Data Memory location within the total available 64 Kbyte space.

```
MOVX  @DPTR,A    ; Load the Data Memory location
                 ; pointed to by the contents of the
                 ; DPTR register with the contents
                 ; of the Accumulator.
```

Immediate Addressing

Immediate Addressing is used to access constants for use as operands which are contained in the current instruction in Program Memory. For example:

```
ORL   A,#040H    ; Logical OR of the Accumulator
                 ; with the constant value of 040H
```

Register-Indirect with Displacement

Register-Indirect with Displacement Addressing is used to access data in look-up tables in Program Memory space. The location accessed is pointed to by the contents of either the DPTR or the PC registers, which are used as a base register added together with the contents of the Accumulator (A), which is used as an index register. As an example:

```
MOVC  A,@DPTR+A  ; Load the Accumulator with
                 ; the contents of the Program
                 ; Memory location pointed to
                 ; by the value of the DPTR
                 ; register plus the value
                 ; contained in the Accumulator
```

Relative Addressing

Relative Addressing is used in the determination of a destination address for the Conditional Branch instructions. Each of these instructions includes an 8-bit byte which contains a 2's complement address offset (-127 to +128) which is added to the PC to determine the destination address which will be branched to when the tested condition is found to be true. The PC points to the Program Memory location immediately after the Branch instruction when the offset is added. If the condition is found to be not true, then program execution continues from the address of the following instruction. As an example:

```
JZ    -20        ; Branch to the location (PC+2) - 20
                 ; if the contents of the Accumulator
                 ; = 0
```

Page Addressing

Page Addressing is used the Control Transfer instructions to specify a destination address within the 2 Kbyte block in which the next contiguous instruction resides. The full 16-bit address is calculated by taking the highest-order five bits for the next contiguous instruction (PC+2) and concatenating them with the lowest-order 11-bit filed contained in the current instruction. 11-bit field provides an efficient instruction encoding of a destination address for these instructions. For example:

```
0830  ACALL   100H   ; Call to the subroutine at
                     ; address 0100H + current
                     ; page address
```

In this case the destination address would be 800H + 100H or 900H.

Extended Addressing

Extended Addressing is used in the Control Transfer Instructions to specify a 16-bit destination address within the entire 64 Kbyte addressable range of the Soft Micro. For example:

```
LJMP  0FF80H     ; Jump to address 0FF80H
```

Program Status Flags

All of the Program Status flags are contained in the PSW register. An operational summary of the relevant bits is given below. Instructions which affect the states of the flags are also summarized below.

PROGRAM STATUS FLAGS

Bit Description:

PSW.7:	**C**
"Carry":	Set when the previous operation resulted in a carry (during addition) or a borrow (during subtraction). Otherwise cleared.
Read/Write:	May be read or written at anytime.
Initialization:	Cleared to a 0 on any type of reset.
PSW.6:	**AC**
"Auxiliary-Carry":	Set when the previous operation resulted in a carry (during addition) or a borrow (during subtraction) from the high-order nibble. Otherwise cleared.
Read/Write:	May be read or written at any time.
Initialization:	Cleared to a 0 on any type of reset.
PSW.2:	**OV**
"Overflow":	Set when a carry was generated into the high-order bit but not a carry out of the high-order bit as a result of the previous operation, and vice-versa. OV is normally used in 2's complement arithmetic.
Read/Write:	May be read or written at any time.
Initialization:	Cleared to a 0 on any type of reset.
PSW.0:	**P**
"Parity":	Set if the module-2 sum of the eight bits of the Accumulator is 1 (odd parity); cleared on even parity.
Read/Write:	May be read at anytime, but is not writable.
Initialization:	Cleared to a 0 on any type of reset.

INSTRUCTIONS THAT AFFECT FLAG SETTINGS

	FLAGS				FLAGS		
INSTRUCTION	C	OV	AC	INSTRUCTION	C	OV	AC
ADD	↕	↕	↕	CLR C	0		
ADDC	↕	↕	↕	CPL C	↕		
SUBB	↕	↕	↕	ANL C, bit	↕		
MUL	0	↕		ANL C, $\overline{bit}$	↕		
DIV	0	↕		ORL C, bit	↕		
DA	↕			ORL C, $\overline{bit}$	↕		
RRC	↕			MOV C, bit	↕		
RLC	↕			CJNE	↕		
SETB C	1						

LEGEND:
0 = Cleared to 0
1 = Set to a 1
↕ = Modified according to the result of the operation.

SECTION 5: MEMORY INTERCONNECT

The Soft Micro line is divided between chips and modules. This sections illustrates the memory interconnect for the various chips and shows block diagrams of selected modules. The Soft Micro chips are 80–pin QFP packages that connect to low power CMOS SRAM. The SRAM connection is made through the Byte–wide bus. When using a chip, the user must connect this Byte–wide bus to the RAM as shown in this section. In module form, the bus is connected inside the package. Table 5–1 shows some of the preferred RAM choices. Note that any standard SRAM will work, but data retention lifetime is dependent on RAM data retention current and battery capacity. Lower currents naturally allow the use of smaller batteries. This is covered in detail in Section 6.

RECOMMENDED SRAMs FOR USE WITH SOFT MICROCONTROLLERS Table 5–1

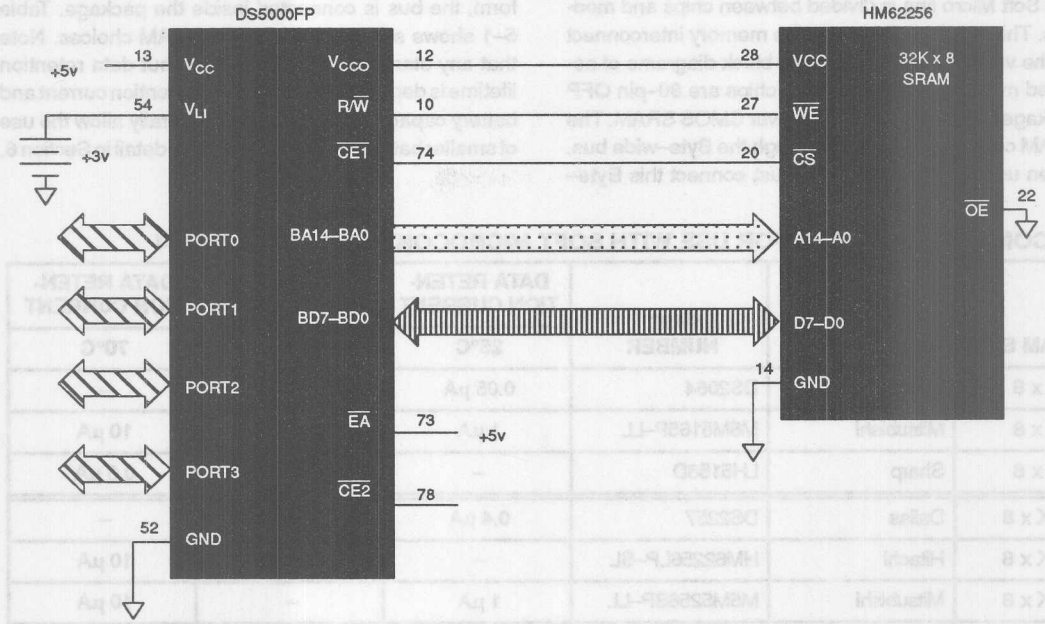
RAM SIZE	VENDOR	PART NUMBER	DATA RETENTION CURRENT 25°C	DATA RETENTION CURRENT 40°C	DATA RETENTION CURRENT 70°C
8K x 8	Dallas	DS2064	0.05 µA	–	–
8K x 8	Mitsubishi	M5M5165P–LL	1 µA	–	10 µA
8K x 8	Sharp	LH5156D	–	–	0.6 µA
32K x 8	Dallas	DS2257	0.4 µA	–	–
32K x 8	Hitachi	HM62256LP–SL	–	3 µA	10 µA
32K x 8	Mitsubishi	M5M5256BP–LL	1 µA	–	10 µA
32K x 8	Sony	CXK58257AP–LX	1 µA	2 µA	10 µA
32K x 8	Sony	CXK58527AP–LLX	0.3 µA	0.6 µA	3 µA
128K x 8	HM628128LP–SL	Hitachi	1 µA	–	10 µA
128K x 8	M5M51008P–LL	Mitsubishi	1 µA	–	10 µA
128K x 8	CXK581000P–LL	Sony	1.2 µA	2.4 µA	12 µA

Recommended RAMs are given with the manufacturers specified data retention current at 3V. Missing numbers are conditions unspecified by the manufacturer.

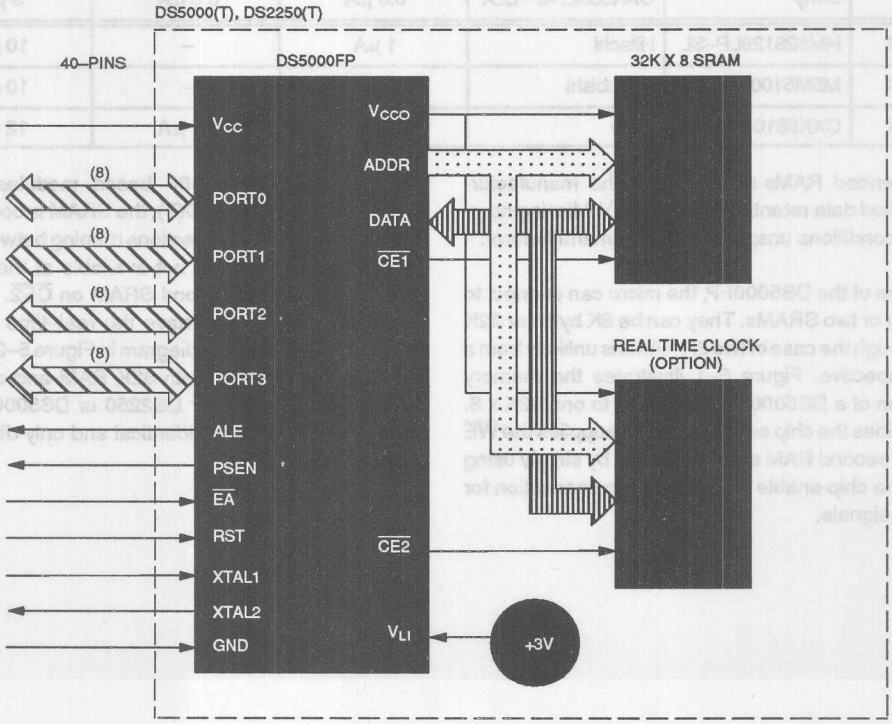
In the case of the DS5000FP, the micro can connect to either one or two SRAMs. They can be 8K bytes or 32K bytes, though the case of two 8K RAMs is unlikely from a cost perspective. Figure 5–1 illustrates the memory connection of a DS5000FP connected to one 32K x 8. $\overline{CE1}$ provides the chip select, and R/$\overline{W}$ supplies the $\overline{WE}$ signal. A second RAM could be added by simply using $\overline{CE2}$ as the chip enable with a common connection for the other signals.

In the case of DS5000 based modules including DS5000(T) and DS2250(T), the SRAM is connected as described above. Connections running between the micro chip and RAM are not available at the pins. The DS2250–64 has a second SRAM on $\overline{CE2}$. The time-keeping versions also have the real–time clock connected to $\overline{CE2}$. A block diagram in Figure 5–2 shows the module configuration with 32K RAM and a real–time clock. This identical for DS2250 or DS5000 modules. These are functionally identical and only differ in form factor.

MEMORY INTERCONNECT OF THE DS5000FP Figure 5–1



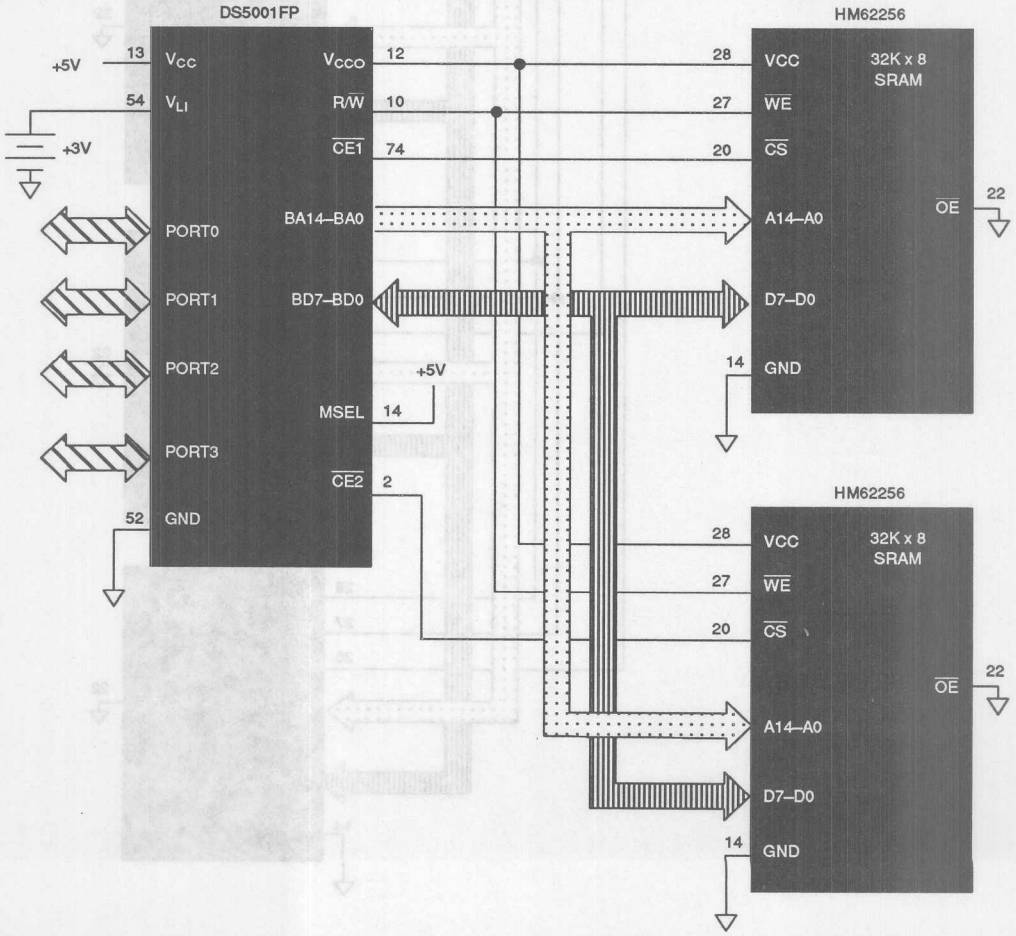
DS5000 SERIES MODULE BLOCK DIAGRAM Figure 5–2



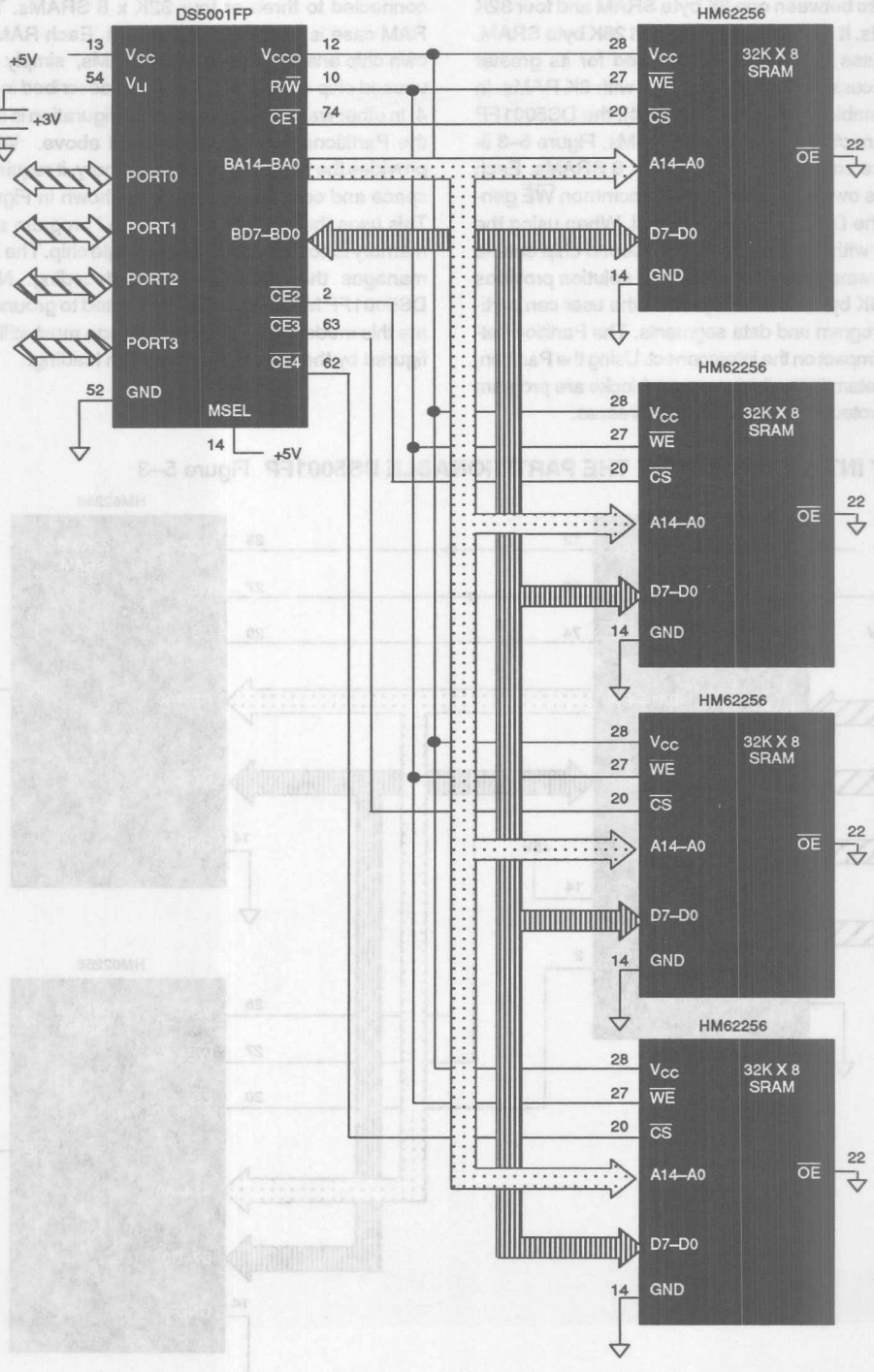
The DS5001FP has several memory options. It can be connected to between one 8K byte SRAM and four 32K byte SRAMs. It will also support one 128K byte SRAM. In most cases the DS5001FP is used for its greater memory access so it will not be used with 8K RAMs. In the Partitionable mode (see Section 4), the DS5001FP can be connected to one or two SRAMs. Figure 5–3 illustrates the connection of two 32K x 8 SRAMs. Each RAM has its own chip enable, with a common $\overline{WE}$ generated by the DS5001FP R/$\overline{W}$ signal. When using the DS5001FP with only one RAM, the second chip enable will simply remain unconnected. This solution provides a total of 64K bytes of memory which the user can partitions into program and data segments. The Partition setting has no impact on the interconnect. Using the Partition, the micro determines which memory blocks are program and write protects the appropriate addresses.

In the non–partitionable case, the DS5001FP can be connected to three or four 32K x 8 SRAMs. The four RAM case is shown in Figure 5–4. Each RAM has its own chip enable. To use three RAMs, simply omit the unused chip enable ($\overline{CE2}$ or 4) as described in Section 4. In other ways, this hardware configuration is similar to the Partitionable mode discussed above. While this provides the full 128K bytes of memory, it requires more space and cost than the version shown in Figure 5–5. This uses the 128K byte SRAM. All program and data memory is contained within the single chip. The DS5001 manages the addressing and decoding. Note the DS5001FP MSEL signal is connected to ground to initiate this mode. The PM bit and Range must still be configured by the user during program loading.

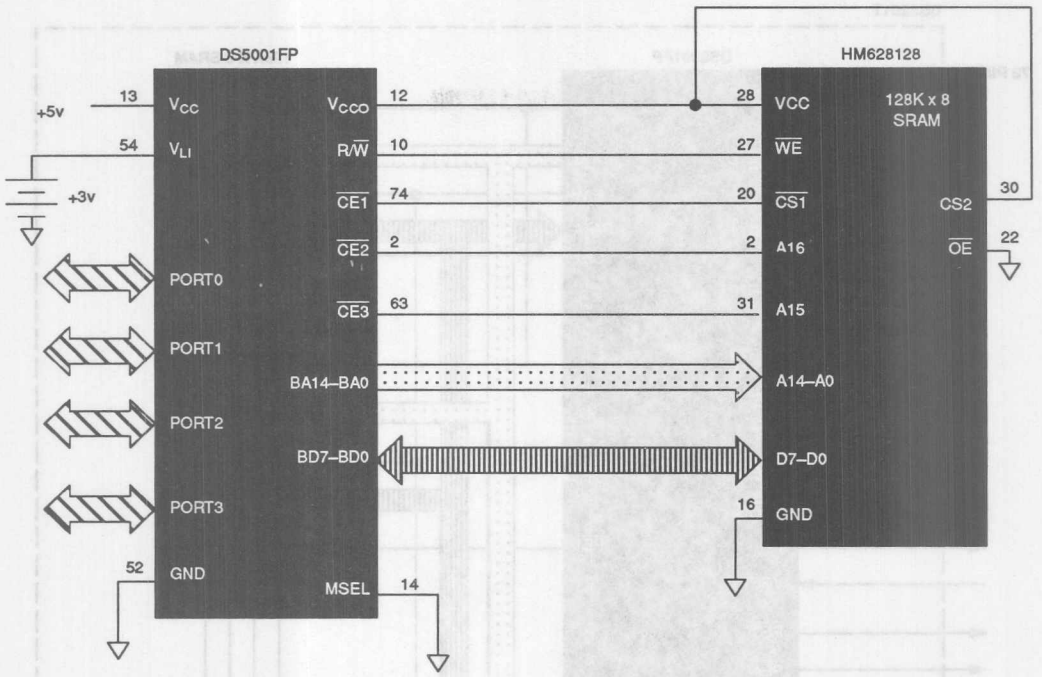
MEMORY INTERCONNECT OF THE PARTITIONABLE DS5001FP Figure 5–3



MEMORY INTERCONNECT OF THE NON–PARTITIONABLE DS5001FP Figure 5–4



MEMORY INTERCONNECT USING THE 128K SRAM Figure 5–5

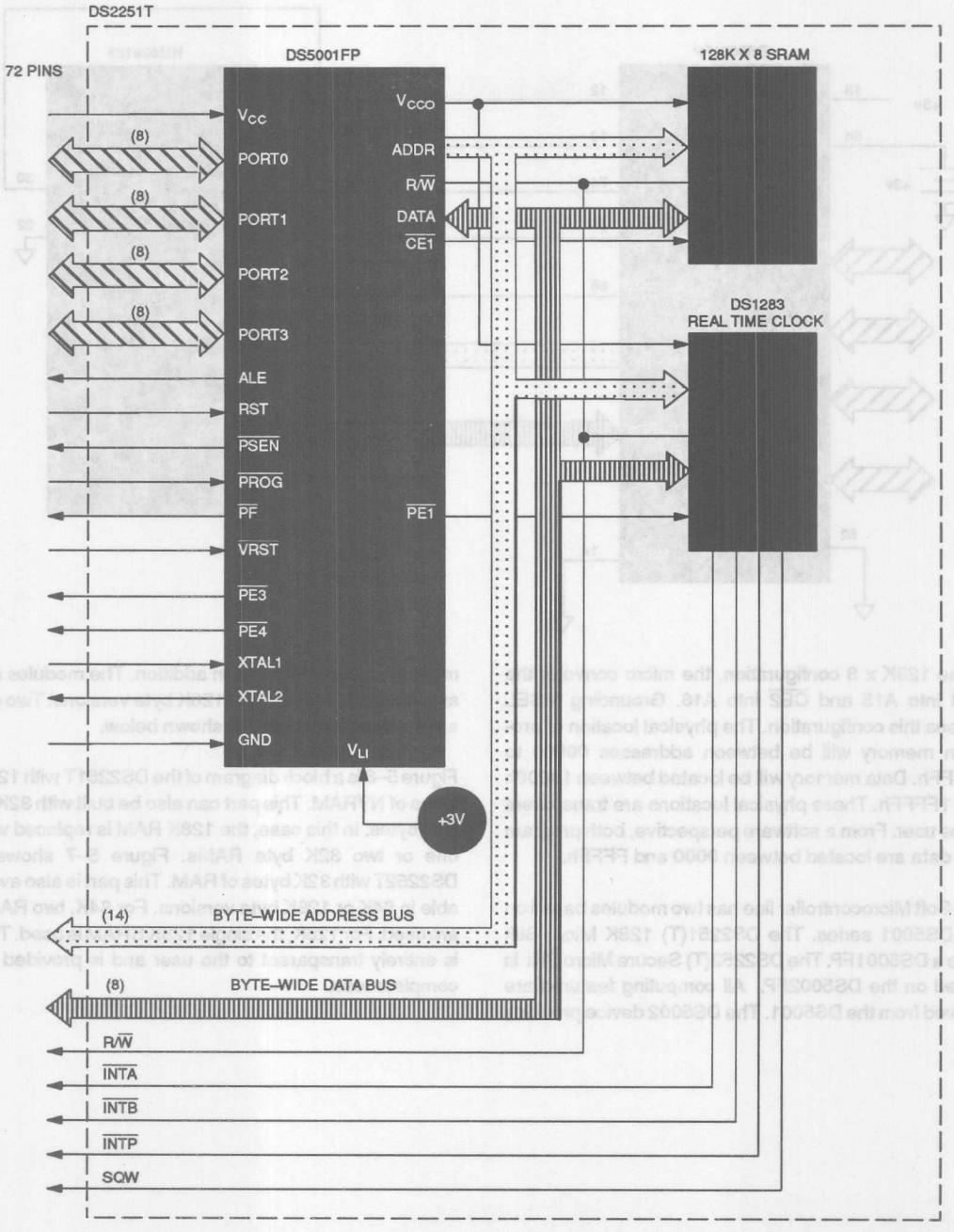


In the 128K x 8 configuration, the micro converts the $\overline{CE3}$ into A15 and $\overline{CE2}$ into A16. Grounding MSEL causes this configuration. The physical location of program memory will be between addresses 00000 to 0FFFFh. Data memory will be located between 10000h and 1FFFFh. These physical locations are transparent to the user. From a software perspective, both program and data are located between 0000 and FFFFh.

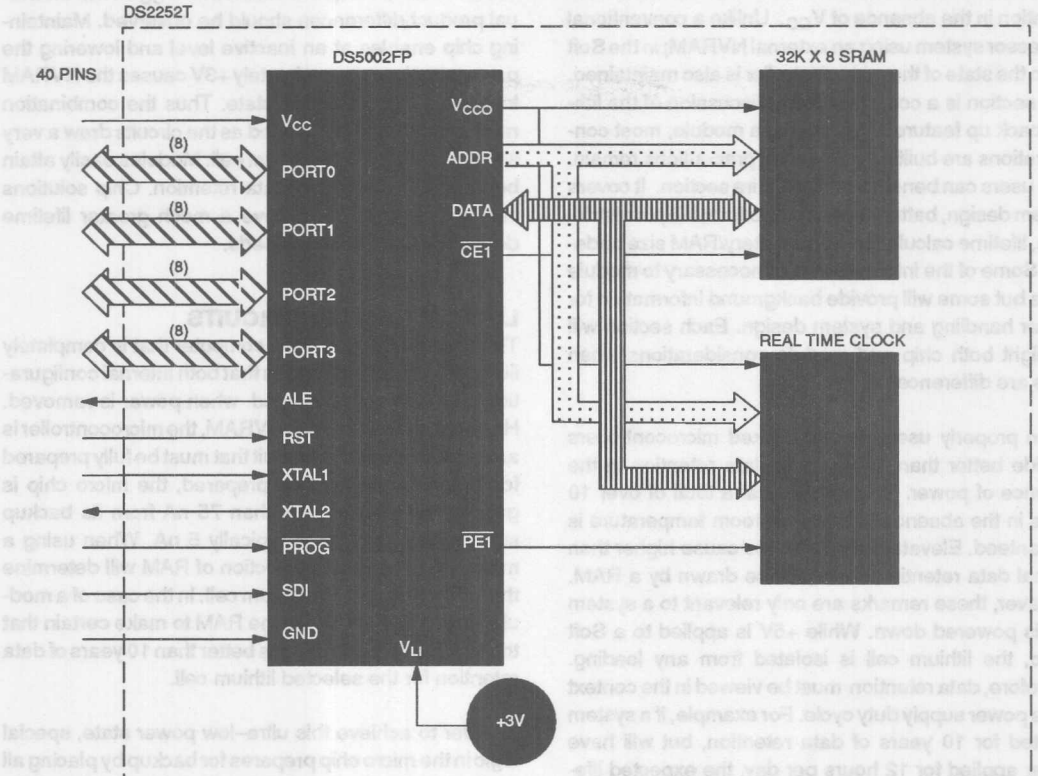
The Soft Microcontroller line has two modules based on the DS5001 series. The DS2251(T) 128K Micro Stik uses a DS5001FP. The DS2252(T) Secure Micro Stik is based on the DS5002FP. All computing features are derived from the DS5001. The DS5002 device provides memory security features in addition. The modules are available in 32K, 64K, and 128K byte versions. Two example block diagrams are shown below.

Figure 5–6 is a block diagram of the DS2251T with 128K bytes of NVRAM. This part can also be built with 32K or 64K bytes. In this case, the 128K RAM is replaced with one or two 32K byte RAMs. Figure 5–7 shows a DS2252T with 32K bytes of RAM. This part is also available in 64K or 128K byte versions. For 64K, two RAMs are used. For 128K, the single 128K SRAM is used. This is entirely transparent to the user and is provided for completeness.

DS2251T–128 BLOCK DIAGRAM Figure 5–6



DS2252T–32 BLOCK DIAGRAM Figure 5–7



SECTION 6: LITHIUM BACKUP

Soft Microcontroller devices are lithium backed for data retention in the absence of V_{CC}. Unlike a conventional processor system using an external NVRAM; in the Soft Micro the state of the microcontroller is also maintained. This section is a comprehensive discussion of the lithium back up feature. When using a module, most considerations are built in, but several precautions remain. Chip users can benefit from the entire section. It covers system design, battery attach procedure, I/O pin restrictions, lifetime calculations, and battery/RAM size tradeoffs. Some of the information is unnecessary to module users but some will provide background information for proper handling and system design. Each section will highlight both chip and module considerations when there are differences.

When properly used, lithium backed microcontrollers provide better than 10 years of data retention in the absence of power. This means that a total of over 10 years in the absence of power at room temperature is guaranteed. Elevated temperatures cause higher than normal data retention current to be drawn by a RAM. However, these remarks are only relevant to a system that is powered down. While +5V is applied to a Soft Micro, the lithium cell is isolated from any loading. Therefore, data retention must be viewed in the context of the power supply duty cycle. For example, if a system is rated for 10 years of data retention, but will have power applied for 12 hours per day, the expected lifetime is greater than 20 years.

DATA RETENTION

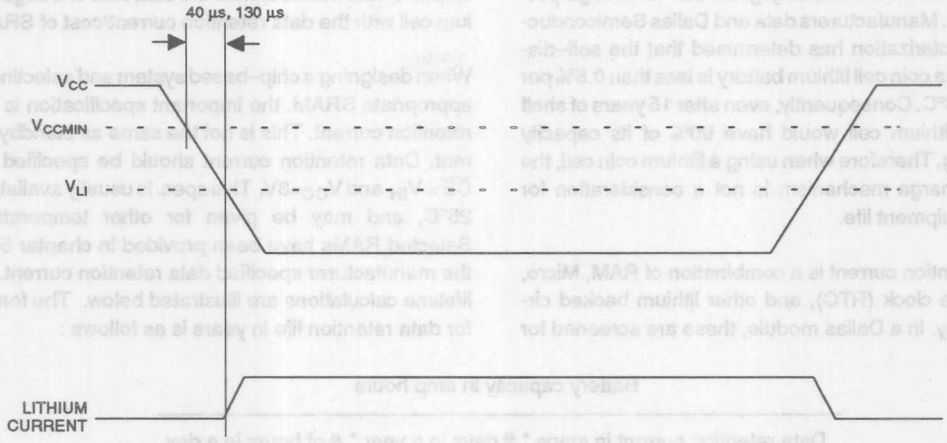
The Soft Microcontroller provides nonvolatile storage in ordinary SRAM. It accomplishes this by lithium backing the memory in the absence of power. When power (V_{CC}) begins to fail, the processor generates an internal power–fail reset condition as discussed in the next chapter. At this time, SRAM chip enables are taken to a logic high inactive state. Also, I/O port pins also go to a logic high state. If power continues to fall and crosses below the lithium threshold, the micro enters the data retention state. Inside the microcontroller, power is drawn from the lithium cell. The power supply output to the SRAM (V_{CCO}) is switched from V_{CC} to the lithium cell. V_{CC} is subsequently ignored, except for comparators that monitor its level. Lithium backed chip enables are maintained at a logic high state with lithium power, but non backed chip enables follow V_{CC} down. Individual product differences should be observed. Maintaining chip enables at an inactive level and lowering the power supply to approximately +3V causes the NVRAM to enter a data retention state. Thus the combination retains data for a long period as the circuits draw a very small current from the lithium cell. Modules easily attain better than 10 years of data retention. Chip solutions can be designed to achieve a much greater lifetime depending on the users needs.

LITHIUM BACKED CIRCUITS

The Soft Micro is the only computer that is completely lithium backed. This means that both internal configuration and data are preserved when power is removed. However, unlike a simple NVRAM, the microcontroller is an extremely complex circuit that must be fully prepared for lithium backup. Once prepared, the micro chip is guaranteed to draw less than 75 nA from its backup source. This number is typically 5 nA. When using a micro chip, the user's selection of RAM will determine the total loading on the lithium cell. In the case of a module, Dallas has screened the RAM to make certain that the total loading guarantees better than 10 years of data retention for the selected lithium cell.

In order to achieve this ultra–low power state, special logic in the micro chip prepares for backup by placing all internal nodes in a predictable (low power) state. This occurs during system power down while V_{CC} is falling below the reset voltage threshold and is still above the lithium voltage. The micro requires a maximum of 40 uS to execute this procedure. Once compete, the micro is ready for lithium backup and will draw almost no power from its backup source. If the power supply slews between these threshold voltages faster than 40 µs (130 µs for DS5001/2), the circuits may not complete the backup procedure and the micro backup current could be substantially greater than 75 nA. Fortunately, a modest amount of system capacitance is enough to prevent fast slewing. The actual value will depend on the total system loading. This slew rate must be adhered to for either a chip or module solution. In either case, the micro must have time to prepare for lithium backup. Figure 6–1 illustrates the power supply conditions that should be met.

POWER SUPPLY SLEW RATE Figure 6–1



Each time V_{CC} is restored, the lithium backed functions will remain as they were left. A result is that many of these values are not altered on a reset condition except for the 'no battery reset'. In the documentation, this is referred to as 'No V_{LI} reset'. This will occur after the first time V_{CC} is applied to the Soft Micro chip. The 'no battery reset' state is documented in the section on resets. A module user will never see the 'no battery reset' condition as it was cleared during assembly and test prior to leaving the factory.

BATTERY ATTACH PROCEDURE

This section applies to Soft Micro chips only. When a Soft Microcontroller chip is received from the factory, it is completely uninitialized. All nonvolatile functions are absent since there is no backup source connected to the chip. As mentioned above, the micro chip must put circuits in a low power state in order to prepare for lithium backup. If a battery were attached to an uninitialized chip, the backup current would be unpredictable. For this reason, the battery attach must be done as follows:

1. Apply V_{CC} to the micro chip.
2. Attach the lithium cell to the V_{LI} input.
3. Configure and program the Soft Micro as normal. (Optional at this time.)
4. Power down the micro (remove V_{CC}) using the guidelines discussed above while leaving the battery attached.

The first time a battery is attached to the micro is a special event. When power is applied in the absence of a lithium cell, the chip performs a No V_{LI} Reset. This allows the micro to initialize control bits that are ordinarily nonvolatile and unaffected by a reset. The silicon will never be completely in this state again unless all power (including battery) is removed by the user. In order to provide the extremely low back up currents (<75 nA), the circuits must configure themselves for lithium backup. This is done when V_{CC} is removed from the chip. That is, the micro chip IS NOT CONFIGURED FOR LITHIUM BACKUP when it is received. Therefore, the battery should be attached with V_{CC} at +5V. This will prevent the micro from placing a load on the lithium cell until V_{CC} is removed. At this time, the micro performs its power down procedure and prepares for ultra low power data retention. Attaching the battery to an unpowered Soft Micro chip places an unknown load on the lithium cell. This may drain the cell excessively and should not be done.

LITHIUM BACKED LIFETIME

The calculations of data retention lifetime are helpful for chip or module users. They can serve as design and system reliability guidelines. All lithium backed microcontroller modules are rated for better than 10 years of data retention in the absence of V_{CC} at 25°C. Following these guidelines, similar performance can be achieved using chips. It is also not difficult to achieve better than 10 years depending on the user's actual environment and design goals.

The system lifetime can be determined from three parameters: 1) Data retention current, 2) Lithium cell

capacity, 3) Lithium self–discharge. Current production lithium cells have extremely good self–discharge performance. Manufacturers data and Dallas Semiconductor characterization has determined that the self–discharge of a coin cell lithium battery is less than 0.5% per year at 25°C. Consequently, even after 15 years of shelf life, the lithium cell would have 90% of its capacity remaining. Therefore when using a lithium coin cell, the self–discharge mechanism is not a consideration for rating equipment life.

Data retention current is a combination of RAM, Micro, Real–time clock (RTC), and other lithium backed circuits if any. In a Dallas module, these are screened for combination with the appropriate battery. In using a chip, the user must balance the size/cost of a larger lithium cell with the data retention current/cost of SRAMs.

When designing a chip–based system and selecting the appropriate SRAM, the important specification is data retention current. This is not the same as standby current. Data retention current should be specified with $\overline{CE} = V_{IH}$ and V_{CC}=3V. This spec. is usually available at 25°C, and may be given for other temperatures. Selected RAMs have been provided in chapter 5 with the manufacturer specified data retention current. The lifetime calculations are illustrated below. The formula for data retention life in years is as follows :

$$\frac{\text{Battery capacity in amp hours}}{\text{Data retention current in amps * \# days in a year * \# of hours in a day}}$$

As an example, the Micro chip is rated for 75 nA, SRAM for 500 nA, RTC for 400 nA for a total of 950 nA. A Panasonic CR1632 lithium cell is used with a capacity of 120 mAh.

$$\frac{120 * 10^{-3}}{(75 + 500 + 400) * 10^{-9} * 24 * 365)} = \frac{120 * 10^{-3}}{8.54 * 10^{-3}} = 14 \text{ years}$$

Thus a system with less than 1 µA of data retention current and a CR1632 lithium cell will achieve well over 10 years of data retention in the absence of V_{CC}. Referring to the recommended RAM chart in the previous section, the user will find a variety of RAMs that allow this at room temperature. It makes no difference if the system operates at 70°C, as long as data retention is at 25°C. If storage is at elevated temperature, than the data retention current should be derated accordingly. If the manufacturer does not specify data retention current over temperature, a conservative number is a 70% increase per 10°C. Thus if a RAM in data retention mode draws 1 µA at 25°C, it will draw approximately 1.7 µA at 35°C. A second example illustrates the case of elevated temperature storage.

In this example, the system is constructed using a DS5001FP chip with a Sony CXK581000P–LL 128K x 8 SRAM. The system will be stored at 40°C. As shown in the table in chapter 5, the data retention current of this RAM is 2.4 µA at 40°C. The DS5001FP data retention current will actually drop as temperature increases, so the maximum of 75 nA is conservative. This gives a total data retention current of 2475 nA. In this system, a Rayovac BR2325 with a capacity of 180 mAh is used.

$$\frac{180 * 10^{-3}}{(2400 + 75) * 10^{-9} * 24 * 365)} = \frac{180 * 10^{-3}}{21.68 * 10^{-3}} = 8.3 \text{ years}$$

Note that these ratings are for data retention so V_{CC} is assumed absent for the entire period. Actual equipment lifetime must take the power supply duty cycle into account.

USE OF LITHIUM CELLS

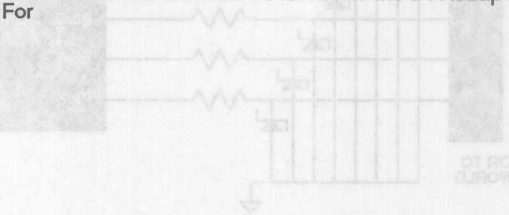
In the vast majority of applications, the lithium cell provides a reliable means of backing up data and configuration. The voltage of lithium cells has only a small variation over its useful life, so it is difficult to measure capacity. A CR chemistry will begin life at 3.3V and drop to 2.9V near the end of life. As a consequence, some users choose to incorporate battery clips so that lithium cells are easily replaced. This is not recommended since such clips are susceptible to shock and vibration. It is possible that the connection to a lithium cell would be momentarily lost during such a shock, resulting in a potential loss of data. Therefore, soldered battery tabs are recommended. If a user elects to use a battery clip with a capacitor (to support momentary disconnect), the leakage of the capacitor should be considered in the lifetime calculations.

FRESHNESS SEAL

The Soft Micro family is designed to maximize the lifetime of a lithium backup source. The circuits described above contribute to a long life. There is one further provision that will benefit users that intend to store their systems in an unpowered state, but that do not require it to retain data during this period. An example might be a completed system stored in inventory. Since data retention is not required, there is no benefit to using even the modest lithium current that will normally be drawn. For this reason, the Soft Micro incorporates the Freshness Seal. The Freshness Seal electrically isolates the lithium cell from any external loading. Thus even in the absence of power, the SRAM and Real–Time Clock leakage currents will not be drawn from the lithium cell for as long as the Freshness Seal is applied.

This feature is available to module users of the DS5000 series [DS5000(T), DS2250(T)] and all users of the DS5001 series [DS5001FP, DS5002FP, DS2251(T), DS2252(T)] In the case of DS5000 and DS2250 modules, the factory ships these with the Freshness Seal applied. In the case of a DS5001 series device, the Freshness Seal can be applied via the Bootstrap Loader at any time. Thus if the Freshness Seal is not removed, the time that a Soft Micro based system is stored in inventory will not reduce the data retention lifetime since the lithium cell is unloaded.

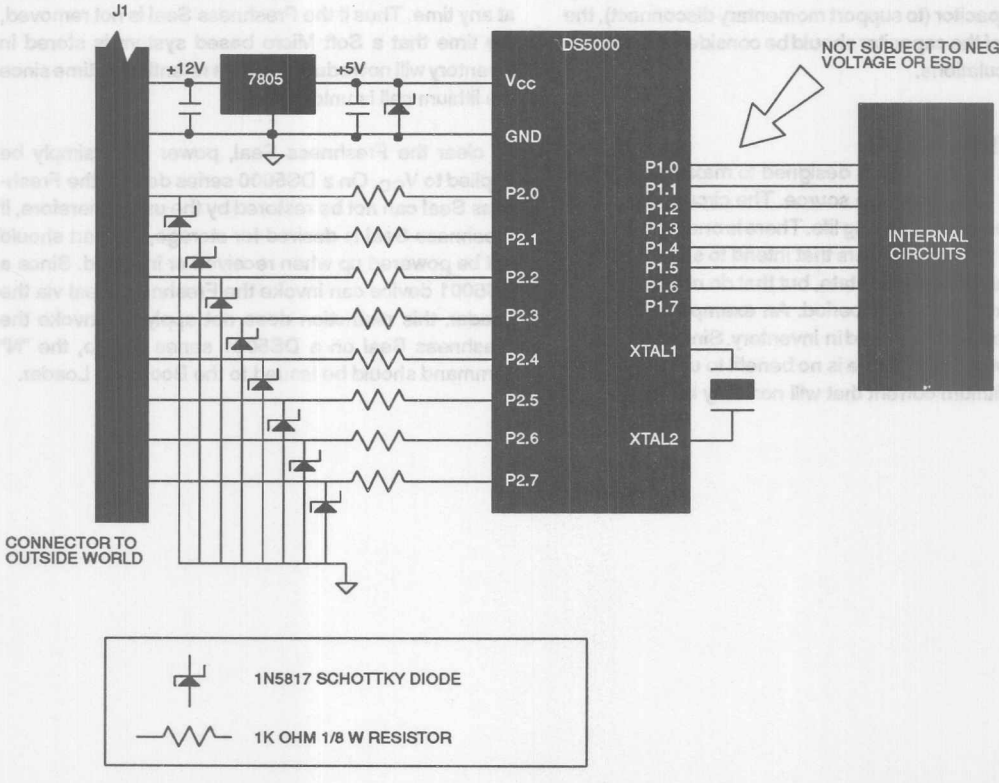
To clear the Freshness Seal, power must simply be applied to V_{CC}. On a DS5000 series device, the Freshness Seal can not be restored by the user. Therefore, if Freshness Seal is desired for storage, the part should not be powered up when received or installed. Since a DS5001 device can invoke the Freshness Seal via the Loader, this restriction does not apply. To invoke the Freshness Seal on a DS5001 series device, the "N" command should be issued to the Bootstrap Loader.



APPLICATION: I/O PIN GUIDELINES

The pins on a Soft Micro chip or module are generally as resilient as other CMOS circuits. They have no unusual susceptibility to ESD or other transients. There is one restriction however. No pin on any Soft Micro device should be taken to a voltage below ground. Taking a pin to a negative voltage turns on internal parasitic diodes that draw current directly from the lithium cell. If a micro pin is connected to the outside world where it may be touched or handled, external reversed–biased Schottky diodes should be used to prevent the potential from going below –0.3V. This voltage is insufficient to activate the internal parasitics. Any on–board circuits than can not be controlled should be treated in a similar manner. Applying a voltage more negative voltage than –0.3V should be avoided. It is also common to see power supplies give a small undershoot on power up. This would constitute a negative voltage and can be treated in the same manner. A reverse–biased schottky diode between V_{CC} and ground will eliminate complications from these undershoots. Figure 6–2 illustrates a circuit designed for a hostile environment.

ESD/NEGATIVE VOLTAGE PRECAUTIONS Figure 6–2



The schottky diode (1N5817) is typical of a 0.3V forward drop silicon diode. The key to using this circuit is that the 1N5817 will turn on (0.1 – 0.2V) well before the internal parasitic diodes (0.5 – 0.6V). Therefore extraneous negative voltage will be clamped to no worse than –0.3V. The current limiting resistor will provide additional protection against fast positive or negative transients. Such protection can be used on any line that is subject to out of spec. voltages, even by accident.

SECTION 7: POWER MANAGEMENT

Introduction

The Soft Micro is implemented using fully static CMOS circuitry for low power consumption. Power consumption is a linear function of crystal frequency. Two software initiated modes are available for further power saving at times when processing is not required and V_{CC} is at normal operating voltage. These are the Idle and Stop modes. The additional third mode offered by the Soft Micro is the Data Retention or Zero Power State which is made possible by the on-chip, crashproof circuitry. The control and status bits which apply to these operating modes are contained in the PCON register and are summarized in Figure 7–1. In addition, Table 7–1 summarizes the state of external pins in each of these modes.

Idle Mode

The Idle mode suspends activity of the CPU. However, the on-chip I/O function, including the timer/counters, and serial port continue their operation. This greatly reduces the number of switching nodes and thereby dramatically reduces the total power consumption of the device. The Idle mode is useful for applications in which lower power consumption is desired with fast response to external interrupts but no other processing.

Software can invoke the Idle mode by setting the IDL bit in the PCON register (PCON.0) to a logic 1 as shown in Figure 7–1. The instruction which sets this bit will be the last instruction executed before Idle mode operation begins. Once in the Idle mode, the Soft Micro preserves the entire CPU status including the Stack Pointer, Program Counter, Program Status Word, Accumulator, and RAM. There are two ways to terminate the Idle mode. The first is from an interrupt which has been previously enabled prior to the Idle mode of operation. This will clear the IDL bit in the PCON register and will cause the CPU to enter the interrupt service routine as normal. When the RETI instruction is executed, the next instruction which will be executed is the one which immediately follows the instruction that set the IDL bit.

The second method of terminating the Idle mode is by a Reset. At this time the IDL bit is cleared and the CPU is placed in the reset state. Since the clock oscillator continues to run in the Idle mode, an oscillator start up delay (referred to as t_{POR} in the AC Electrical Specifications) will not be generated following the reset. Two machine cycles are required to complete the reset operation (24 oscillator periods). It should be noted that the Watchdog Timer continues to run during Idle and that a reset from the on-chip Watchdog Timer will terminate Idle mode.

CONTROL/STATUS BITS FOR POWER CONTROL Figure 7–1

Bit Description:

PCON.6:	**$\overline{POR}$**
"Power On Reset"	Indicates that the previous reset was initiated during a Power On sequence.
Initialization:	Cleared to a 0 when a Power On Reset occurs. Remains at 0 until it is set to a 1 by software.
Read Access:	Can be read normally at any time.
Write Access:	Can be written only by using the Timed Access register.
PCON.5:	**PFW**
"Power Fail Warning"	Indicates that a potential power failure is in progress. Set to a 1 when V_{CC} voltage is below the V_{PFW} threshold. Cleared to a 0 immediately following a read of the PCON register. Once set, it will remain set until read regardless of V_{CC}.
Initialization:	Cleared to a 0 during a Power–On Reset.
Read Access:	Can be read normally at any time.
Write Access:	Cannot be written.

PCON.3: **EPFW**

"Enable Power Fail Interrupt": Used to enable or disable the Power Fail Interrupt. When EPFW is set to a 1, it will be enabled; it will be disabled when EPFW is cleared to a 0.

Initialization: Cleared to a 0 on any type of reset.

Read Access: Can be read normally anytime.

Write Access: Can be written normally anytime.

PCON.1: **STOP**

"Stop": Used to invoke the Stop mode. When set to a 1, program execution will terminate immediately and Stop mode operation will commence. Cleared to a 0 when program execution resumes following a hardware reset.

Initialization: Clear to a 0 on any type of reset.

Read Access: Can be read anytime.

Write Access: Can be written only by using the Timed Access register.

PCON.0: **IDL**

"Idle": Used to invoke to Idle mode. When set at a 1, program execution will be halted and will resume when the Idle bit is cleared to 0 following an interrupt or a hardware reset.

Initialization: Cleared to 0 on any type of reset or interrupt.

Read Access: Can be read normally anytime.

Write Access: Can be written normally anytime.

PIN STATES IN STANDBY MODES Table 7–1

MODE	PROGRAM MEMORY	ALE	PSEN	P0	P1	P2	P3
Idle	Byte–wide	1	1	Port Data	Port Data	Port Data	Port Data
Idle	Expanded	1	1	Hi-Z	Port Data	Address	Port Data
Stop	Byte–wide	1	0	Port Data	Port Data	Port Data	Port Data
Stop	Expanded	1	0	Hi-Z	Port Data	Port Data	Port Data

Stop Mode

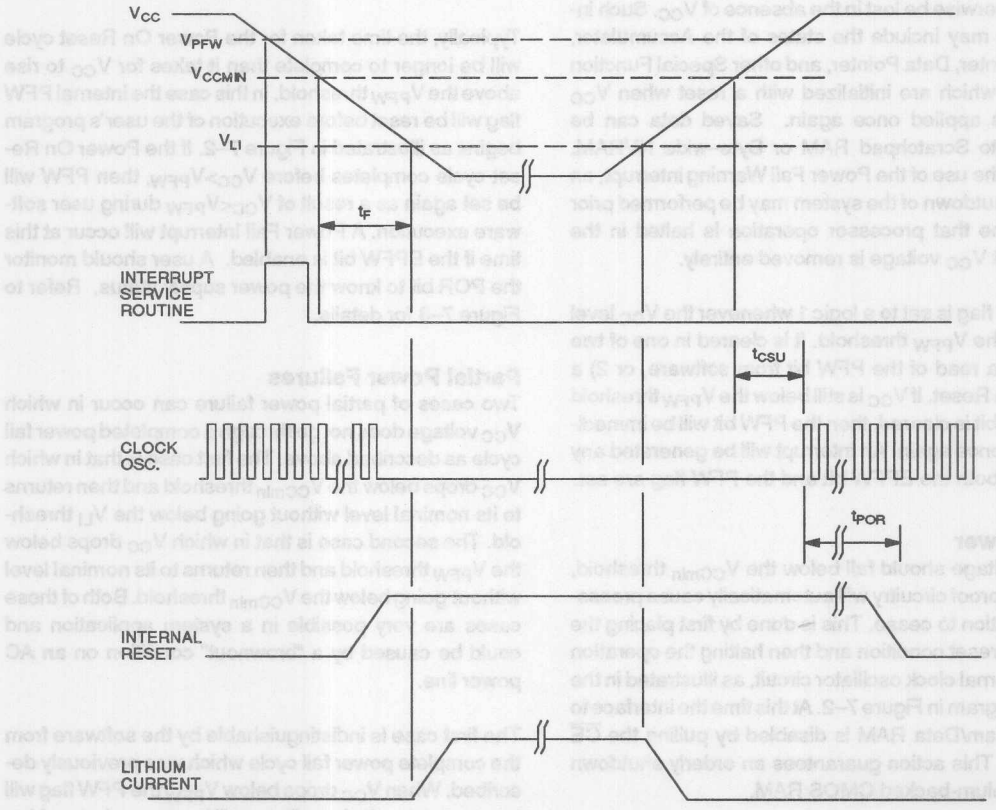
The Stop mode is initiated by setting the STOP bit in the PCON register (PCON.1). The operation of the oscillator is halted in the Stop mode so that no internal clocking signals are produced for either the CPU or the I/O circuitry. An External Reset via the RST pin is the only means of exiting this mode without powering down (V_{CC} taken below V_{CCmin}) and then back up to produce a Power On Reset. The STOP bit may only be set by using the Timed Access software procedure described in Section 8. Since the oscillator is disabled in this mode, the Watchdog Timer will cease operation. When the external reset signal is issued to terminate the Stop mode, a 21,504 clock delay will be generated to allow the clock oscillator to start up and its frequency to stabilize as is done for a Power On Reset as described in Section 10. The original contents of those Special Function registers that are initialized by a reset are lost.

Crashproof Circuitry

The on-chip crashproof circuitry automatically places the Soft Micro in its Data Retention state in the absence of V_{CC}. It insures that the proper internal control signals are generated and that power from the lithium cell is applied at the proper times so that the Program/Data RAM, data in the Scratchpad Registers and certain Special Function Registers remain unchanged when V_{CC} is cycled on and off. In addition, an interrupt is available for signaling the processor of an impending power fail condition so that the operational state of the processor can be saved just prior to entering the Data Retention.

V_{CC} is monitored by the crashproof circuitry for three voltage thresholds below nominal operating voltage. These thresholds are identified as V_{PFW} (Power Fail Warning voltage), V_{CCmin} (minimum operating voltage), and V_{LI} (lithium supply) voltage. These thresholds are used to initiate required actions within the Soft Micro during situations when V_{CC} power is cycled on and off. The timing diagram shown in Figure 7–2 illustrates key internal activities during power cycling.

SOFT MICRO POWER CYCLING TIMING Figure 7–2



Power Fail Interrupt

When V_{CC} is stable, program execution proceeds as normal. If V_{CC} should decay from its nominal operating voltage and drop to a level below the V_{PFW} threshold, then the internal PFW status flag (PCON.5) will be set. In addition, a Power Fail Warning interrupt will be generated if it has been enabled via the EPFW control bit (PCON.3). The purpose of these indicators is to warn the processor of a potential power failure.

The V_{PFW} threshold is above the specified minimum value for V_{CC} (V_{CCmin}) for full processor operation. The V_{PFW} threshold is selected so that with a reasonable power supply slew rate, ample time is allowed for the application software to save all critical information which would otherwise be lost in the absence of V_{CC}. Such information may include the states of the Accumulator, Stack Pointer, Data Pointer, and other Special Function registers which are initialized with a reset when V_{CC} voltage is applied once again. Saved data can be placed into Scratchpad RAM or Byte-wide NVRAM. Through the use of the Power Fail Warning interrupt, an orderly shutdown of the system may be performed prior to the time that processor operation is halted in the event that V_{CC} voltage is removed entirely.

The PFW flag is set to a logic 1 whenever the V_{CC} level is below the V_{PFW} threshold. It is cleared in one of two ways: 1) a read of the PFW bit from software, or 2) a Power On Reset. If V_{CC} is still below the V_{PFW} threshold when the bit is cleared, then the PFW bit will be immediately set once again. An interrupt will be generated any time that both the EPFW bit and the PFW flag are set.

Zero Power

If V_{CC} voltage should fall below the V_{CCmin} threshold, the crashproof circuitry will automatically cause processor operation to cease. This is done by first placing the CPU in a reset condition and then halting the operation of the internal clock oscillator circuit, as illustrated in the timing diagram in Figure 7–2. At this time the interface to the Program/Data RAM is disabled by pulling the $\overline{CE}$ line high. This action guarantees an orderly shutdown for the lithium-backed CMOS RAM.

If V_{CC} voltage drops below V_{LI}, then the Soft Micro is automatically placed in the Data Retention state. The control circuitry accomplishes this by switching the internal power supply line (V_{CCI}) from pin to the lithium power source. At this time, data is retained and no power is drawn from V_{CC}.

When power is once again applied to the system, the V_{CC} voltage will eventually cross the V_{LI} threshold. When this action is detected, the Soft Micro will automatically switch its internal supply line from the lithium source back to the V_{CC} pin. When V_{CC} voltage eventually goes above the V_{CCmin} threshold, the clock oscillator is allowed to start up and an internal Power On Reset cycle is executed. Part of the cycle involves a considerable delay that is generated to allow the clock oscillator frequency to stabilize. Activity on the RST pin is ignored until this sequence is completed. The time required for this cycle is shown as t_{POR} in Figure 7–2 and is specified in the AC Electrical Specifications. A detailed description of the Power On reset cycle operation is given in Section 10.

Typically, the time taken for the Power On Reset cycle will be longer to complete than it takes for V_{CC} to rise above the V_{PFW} threshold. In this case the internal PFW flag will be reset before execution of the user's program begins as illustrated in Figure 7–2. If the Power On Reset cycle completes before $V_{CC}>V_{PFW}$, then PFW will be set again as a result of $V_{CC}<V_{PFW}$ during user software execution. A Power Fail Interrupt will occur at this time if the EPFW bit is enabled. A user should monitor the POR bit to know the power supply status. Refer to Figure 7–3 for details.

Partial Power Failures

Two cases of partial power failure can occur in which V_{CC} voltage does not go through a completed power fail cycle as described above. The first case is that in which V_{CC} drops below the V_{CCmin} threshold and then returns to its nominal level without going below the V_{LI} threshold. The second case is that in which V_{CC} drops below the V_{PFW} threshold and then returns to its nominal level without going below the V_{CCmin} threshold. Both of these cases are very possible in a system application and could be caused by a "brownout" condition on an AC power line.

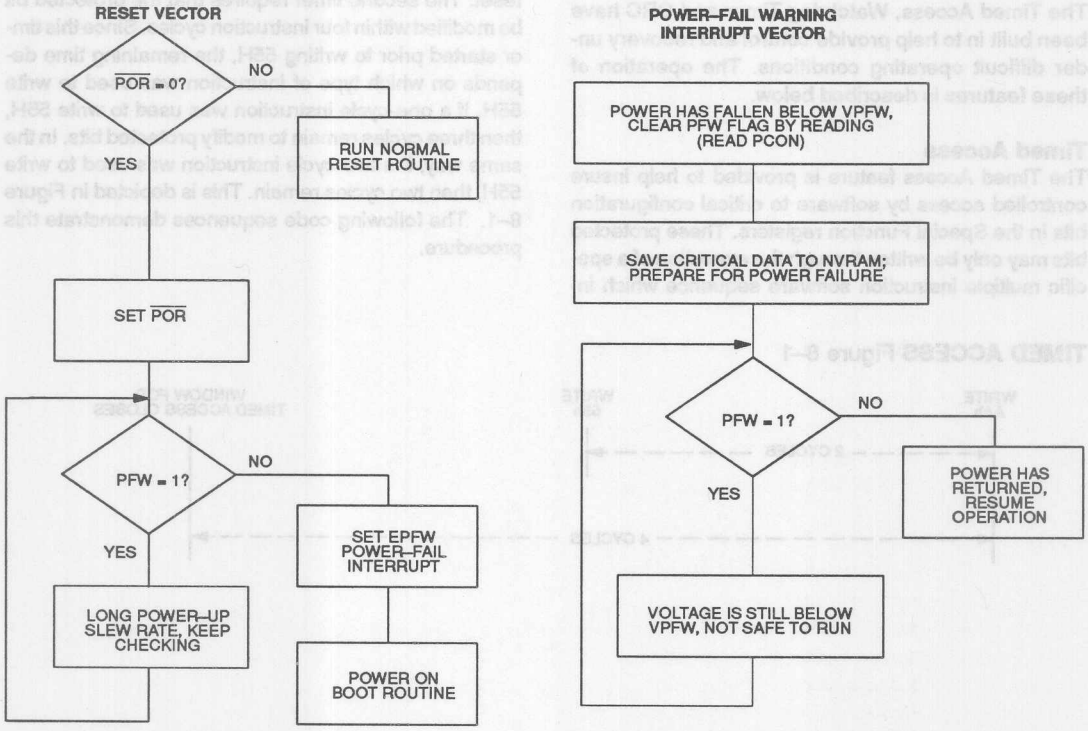
The first case is indistinguishable by the software from the complete power fail cycle which was previously described. When V_{CC} drops below V_{PFW} the PFW flag will be set and the clock oscillator will be stopped when V_{CC} drops below V_{CCmin}. The only operational difference is that if V_{CC} never drops below the V_{LI} threshold, the internal power supply line will never be switched over to the lithium cell. When V_{CC} rises back above the V_{CCmin} threshold, the Power On Reset cycle will be executed as

before. As a result, no special processing is required in software to accommodate this case.

In the second case the PFW flag will be set and a Power Fail Warning interrupt will still occur when V_{CC} drops below the V_{PFW} threshold. The PFW flag will remain set until it is cleared by either a reset of the flag by the software or by a Power On cycle. If it is cleared while V_{CC} is still below the V_{PFW} threshold, it will be immediately set again. If it is cleared after V_{CC} has risen back above the V_{PFW} threshold, then it will remain cleared until the next time V_{CC} goes below V_{PFW}.

As long as the PFW flag is set, an interrupt condition is defined if EPFW is set. If the software executes a service routine in response to a PFW interrupt and exits the service routine with the PFW flag still set, then the processor will be immediately interrupted again. In a typical application, however, the Power Fail Interrupt service routine would test the PFW flag in a conditional loop to determine if V_{CC} has risen back above V_{PFW} and would then return control to the main program in response to the event. See Figure 7–3 for details.

SOFT MICRO POWER MANAGEMENT Figure 7–3



SECTION 8: SOFTWARE CONTROL

Introduction

Several features have been incorporated into the Soft Micro to help insure the orderly execution of the application software in the face of harsh electrical environments. Any microcontroller which is operating in a particularly noisy environment is susceptible to loss of software control. Electrical transients such as a glitch on the clock or a noise spike on an I/O pin can cause software problems like the loss of key variables in internal registers and/or execution of code out of its logical sequence. Such transients can send the microcontroller into an indefinite period of seemingly random software execution.

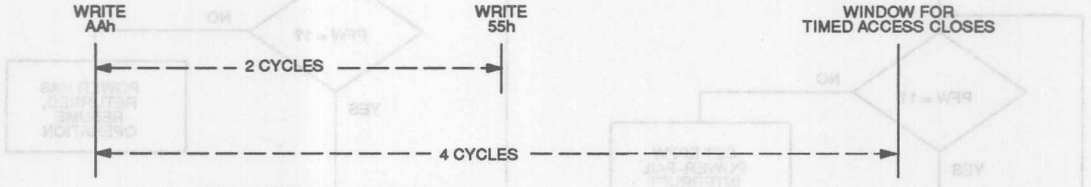
The Timed Access, Watchdog Timer and CRC have been built in to help provide control and recovery under difficult operating conditions. The operation of these features is described below.

Timed Access

The Timed Access feature is provided to help insure controlled access by software to critical configuration bits in the Special Function registers. These protected bits may only be written through the execution of a specific multiple instruction software sequence which involves the Timed Access register. This restriction is designed to help prevent a potentially catastrophic change in the configuration by an inadvertent write during times when software control has been lost.

In order to modify the protected bits listed in Table 8–1, a pattern of two bytes must first be written to the Timed Access register at location 0C7h. The first write should be a value of 0AAh and the second should be a value of 55H. After this sequence is performed, the protected bits may be modified. Upon receiving a 0AAH in the Timed Access register, two timers are initiated. The first timer allows two instruction cycles to write a 55H. This means a one- or two-cycle instruction may be used. If 55H is not written within two cycles, Timed Access is reset. The second timer requires that the protected bit be modified within four instruction cycles. Since this timer started prior to writing 55H, the remaining time depends on which type of instruction was used to write 55H. If a one-cycle instruction was used to write 55H, then three cycles remain to modify protected bits. In the same way, if a two-cycle instruction was used to write 55H, then two cycles remain. This is depicted in Figure 8–1. The following code sequences demonstrate this procedure.

TIMED ACCESS Figure 8–1



This code allows the reset of the Watchdog Timer:

```
MOV     0C7H,#0AAH      ; 1st TA Value
MOV     0C7H,#055H      ; 2nd TA Value          2 Cycles
SETB    IP.7            ; Reset Watchdog Timer  1 Cycle
```

The Watchdog Timer bit may have been set using ORL IP,#80H which takes two cycles.

This code allows the reset of the Watchdog Timer using a different approach:

```
MOV     A,#55H          ; Setup Acc for fast write
MOV     0C7H,#0AAH      ; 1st TA Value
MOV     0C7H,A          ; 2nd TA Value          1 Cycle
MOV     A,IP            ; Get Current IP        1 Cycle
ORL     A,#80H          ; Prepare for fast write 1 Cycle
MOV     IP,A            ; Reset Watchdog Timer  1 Cycle
```

Note that a new value for IP could have been retrieved from any direct register instead of the current IP.

The bits which are write access-protected by the Timed Access function are listed in Table 8–1.

TIMED ACCESS PROTECTED CONTROL BITS Table 8–1

BIT NAME	MICRO VERSION	LOCATION	DESCRIPTION
EWT	All Soft Micro	PCON.2	Enables the Watchdog Timer Reset function
RWT	All Soft Micro	IP.7	Resets the Watchdog Timer count
STOP	All Soft Micro	PCON.1	Stop Mode Enable
POR	All Soft Micro	PCON.6	Power On Reset
PAA	DS5000 series	MCON.1	Partition Address Access bit (protects PA3–0)
PA3–0	DS5001 series	MCON.7–4	Partition Address bits
AE	DS5001 series	RPCTL.4	Access Enable

The Soft Micro has a variety of control bits that are critical to the correct operation of the processor. Several of these are nonvolatile and will not be altered by a reset. Thus they must be protected from an accidental write by software that has gone out of control. This is a possibility in all microprocessor based systems, especially those in an industrial environment. While the Watchdog Timer will recover from this condition, the critical bits must be protected during the interval before the time–out of the Watchdog Timer.

The Soft Micro actually has two levels of protection for these critical bits. The most critical SFR bits can only be altered using the Bootstrap Loader. These can not be altered by the application software. An example is the Range function that determines the total memory. This will not change so the application software would not need to modify it. For bits that might need to change but which are critical to crashproof operation, the Timed Access procedure is used to protect against an inadvertent write operation. Timed Access provides a statistical protection. It is unlikely that randomly generated states will correctly match the sequence and timing required to bypass the Timed Access logic. Presented below is a brief justification for each bit that is protected by Timed Access.

The EWT bit is protected in order to prevent errant software from disabling the Watchdog Timer. The Watchdog is one of the important mechanisms that assure correct operation and should not be turned off accidentally. RWT is the bit that software uses to restart the Watchdog time–out. The Soft Micro makes this more difficult by Timed Access protecting the bit. Thus software must "really" intend to reset the time–out in order to do so. STOP mode is the lowest power state that is available while V_{CC} is applied. To gain this low power condition, all clocks in the micro are stopped. Even the Watchdog Timer is turned off. Thus software should not accidentally place the micro in STOP mode where the Watchdog could not recover.

POR informs the software of the power supply condition. Specifically, it means the power has previously dropped below the V_{CCMIN} level and returned to normal. In many systems, this is a unique condition that requires interaction with external hardware. Protecting this bit with a Timed Access procedure prevents the micro from accidentally performing a power on reset procedure.

On a DS5000 series device, the PAA bit allows software to alter the Partition. If this is done accidentally, the resulting configuration could be unrecoverable without human intervention. This could mean selecting a Partition that is outside of the user's plan and that causes the system to fail. In a like manner, the PA3–0 bits on a DS5001 series device are protected through Timed Access. As the DS5001 does not have a PAA bit, the Partition control bits are directly protected. The motivation for protecting the AE bit is similar. This bit invokes a Partitionable configuration where one had not been selected during Bootstrap loading. While there are several valid reasons to select AE, accidentally selecting this condition might be unrecoverable without manual intervention.

Note that the Timed Access logic protects against the possibility of a single inadvertent write modifying a critical control bit. It does not protect against inadvertently entering a section of code that contains the correct sequence to modify a protected bit. However, the statistical protection does greatly improve the system's resilience to a crash.

Watchdog Timer

The on-chip Watchdog Timer provides a method of restoring proper operation during transients that cause the loss of controlled execution of software. When the Watchdog Timer is enabled, it will eventually reach a timeout condition after 122,800 machine cycles unless it is reset by the application software. An internal reset to the CPU will be generated if the timeout condition is ever reached. Application software which utilizes the Watchdog Timer should therefore be written to periodically reset it so that the timeout condition will never be reached during normal operation. The reset operation(s) should be inserted at critical check points in the program which under normal operation, should be reached before the timeout period has expired. The Watchdog Timer will monitor program execution to insure that these check points are reached, indicating proper operation. If controlled execution of the software is lost so that these check points in the program are not encountered within the timeout period, then the Watchdog Timer will provide an automatic reset. A block diagram of the Watchdog Timer is shown in Figure 8–2.

The Special Function Register bits that are used to control the Watchdog include the Enable Watchdog Timer bit (EWT; PCON.2), the Reset Watchdog Timer bit (RWT; IP.7), and the Watchdog Timer Reset status flag (WTR; PCON.4). The Watchdog Timer incorporates a free-running counter that starts counting as soon as the clock oscillator begins operation following a Power On Reset. If a 12 MHz crystal is used as the time base element, this gives a timeout period of 122.88 ms. The Watchdog Timer Reset function is enabled with a Timed Access write operation which sets the EWT bit to a 1. A Watchdog Timer Reset will then occur the next time that the free-running counter reaches its timeout condition.

Regardless of whether the Watchdog Timer will be used, it should be initialized after each reset. If the Watchdog Timer is desired, then the first step is to reset the timer count. This is necessary since the timer is free running and may be about to time–out. Set the RWT bit to a logic 1 using a Timed Access procedure. This will restart the timer with the full interval. Then enable the Watchdog Timer reset function by setting the EWT bit to a logic 1, again with a Timed Access procedure. Note that the EWT bit only controls whether the reset is issued, not whether the timer runs. The Watchdog Timer must now be reset prior to 122,800 machine cycles or it will reset the CPU. If the Watchdog Timer is not used, then clear the EWT bit to a logic 0 using a Timed Access procedure. Since the EWT bit is nonvolatile, this makes certain that the Watchdog reset function remains disabled.

During subsequent program execution, the Watchdog Timer can be reset by a Timed Access write operation which sets the RWT bit to a 1. This will cause the Watchdog Timer to begin counting machine cycles again from an initial count of 0. The RWT bit itself is automatically cleared immediately after the Watchdog Timer is reset. An instruction sequence which performs this operation is as follows.

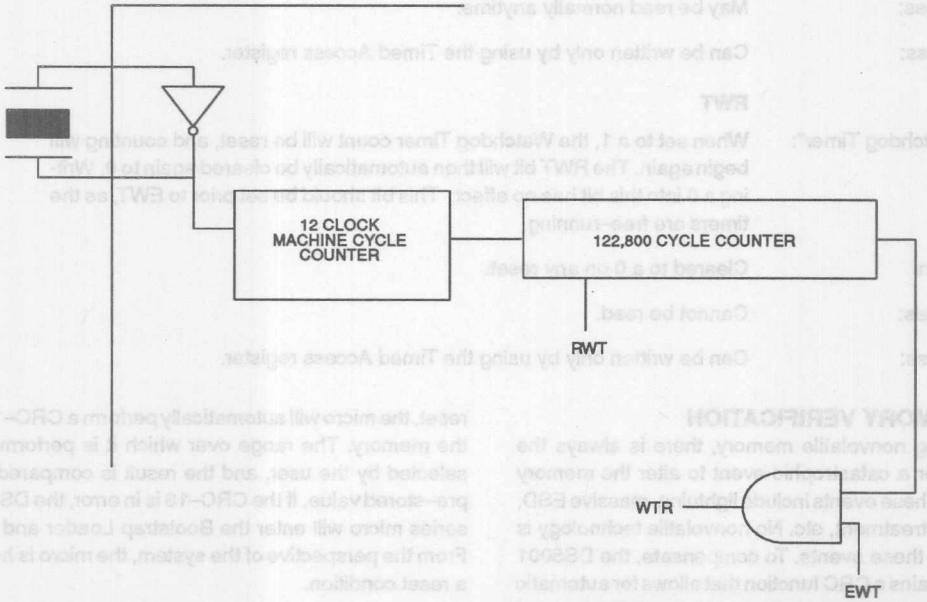
This code allows the reset of the Watchdog Timer:

```
MOV   0C7H,#0AAH   ; 1st TA Value
MOV   0C7H,#055H   ; 2nd TA Value
SETB  IP.7         ; Reset Watchdog Timer
```

If the timeout period is ever reached without the timer being reset by the software, the Watchdog Timer will reset the CPU, set the WTR status flag, and will begin counting again. The WTR flag allows the application software to distinguish this type of reset from other possible sources so that special processing can be performed to accommodate this case. This flag will be set in response to a timeout, regardless of whether the reset is enabled. The WTR bit is cleared only by a read of the PCON register. Therefore, this register should be read during initialization following a reset in order to properly interpret the source of the reset. The Watchdog Timer is also reset whenever any other type of reset is issued to the CPU and will begin its count as soon as the reset condition is released and the application software begins execution.

If operation without the Watchdog Timer is desired, then the EWT bit should be cleared following any type of reset by using the Timed Access register. This will insure that the Watchdog Timer will never cause an undesired reset during execution of the application software.

WATCHDOG TIMER Figure 8–2



WATCHDOG TIMER CONTROL BITS

Bit Description:

PCON.4:	**WTR**
"Watchdog Timer Reset"	Set to a 1 when a Watchdog Timer timeout occurs. If Watchdog Timer Reset is enabled, this will indicate the cause of the reset. Cleared to 0 immediately following a read of the PCON register.
Initialization:	Set to a 1 after a Watchdog Timeout. Cleared to a 0 on a No-V_{LI} Power On Reset. Remains unchanged during other types of resets.
Read Access:	May be read normally anytime.
Write Access:	Cannot be written.
PCON.2:	**EWT**
"Enable Watchdog Timer Reset":	Used to enable or disable the Watchdog Timeout Reset. The Reset is enabled if EWT is set to a 1 and will be disabled if EWT is cleared to a 0. This bit affects the generation of a reset condition, not the running of the Watchdog Timer.
Initialization:	Cleared to a 0 on a No-V_{LI} Power On Reset. Remains unchanged during other types of resets.
Read Access:	May be read normally anytime.
Write Access:	Can be written only by using the Timed Access register.
IP.7:	**RWT**
"Reset Watchdog Timer":	When set to a 1, the Watchdog Timer count will be reset, and counting will begin again. The RWT bit will then automatically be cleared again to 0. Writing a 0 into this bit has no effect. This bit should be set prior to EWT, as the timers are free–running.
Initialization:	Cleared to a 0 on any reset.
Read Access:	Cannot be read.
Write Access:	Can be written only by using the Timed Access register.

CRC MEMORY VERIFICATION

When using nonvolatile memory, there is always the potential for a catastrophic event to alter the memory contents. These events include lightning, massive ESD, severe mistreatment, etc. No nonvolatile technology is immune to these events. To compensate, the DS5001 series contains a CRC function that allows for automatic verification of memory on power up. The CRC function is also available to the user for application software use. Note that this is not available on DS5000 series devices [DS5000(T), DS2250(T), DS5000FP].

If the CRC option is selected through the Bootstrap Loader, then on power up or after a Watchdog Timer reset, the micro will automatically perform a CRC–16 on the memory. The range over which it is performed is selected by the user, and the result is compared to a pre–stored value. If the CRC–16 is in error, the DS5001 series micro will enter the Bootstrap Loader and wait. From the perspective of the system, the micro is held in a reset condition.

To support this function, the CRC register shown below is accessible through the Bootstrap Loader. Setting the CRC bit (LSB) enables the power–up CRC function. The loader command "W" is used to write to this register. The upper nibble of the CRC register (a hex value between 0 and F) defines the address space in 4K

blocks over which the CRC calculation is performed. For example, if the nibble is set to 0001b, the CRC range is from 0000 to 0FFFh. Once the LSB of the CRC register is set, the loader "I" command will cause the CRC of the specified block to be computed. The result is automatically stored in the last two bytes of the specified block. These bytes should not be used by the application. This computation will be correct provided that the CRC range is less than or equal to the partition if PM=0. If PM=1, using 32K RAMs, the CRC range must be less than or equal to the program range.

If CRC is enabled, the DS5001FP will automatically invoke the Bootstrap Loader on either power–up or a Watchdog timeout and the CRC check will be performed. If an error is detected, the Bootstrap Loader will wait for reloading. If there is no error, the application will begin at address 0000h with a reset. Automatic checking of the CRC can be disabled by writing a 0 to the CRC register LSB. As mentioned above, this is done using the "W" command in loader mode. The CRC hardware uses registers 0C3h and 0C2h for most and least significant byte intermediate storage.

CRC REGISTER (Address 0C1h)

RNGE3	RNGE2	RNGE1	RNGE0	—	—	MDM	CRC

CRC.7–4: **RANGE 3–0**
Determines the range over which a power–up CRC will be performed. Addresses are specified on 4K boundaries.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by application software. Can be written via the Bootstrap Loader.

CRC.1: **MDM**
When set to 1, the Bootstrap Loader will attempt to use a modem (UART) on $\overline{PE4}$ if CRC is incorrect.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by application software. Can be written via the Bootstrap Loader.

CRC.0: **CRC**
When set to 1, a CRC check will be performed on power–up or watchdog timeout. CRC will be checked to stored values. An error will initiate program load mode.

Initialization: Reset to 0 on a No V_{LI} reset.

Read Access: Can be read at any time.

Write Access: Cannot be written by application software. Can be written via the Bootstrap Loader.

CRC CODE EXAMPLE Figure 8–3

```
                This routine tests the CRC–16 circuit in the DS5001FP

crcmsb      equ     0C3h
crclsb      equ     0C2h

            org     00h         ;after reset, CRC regs = 0000
begin:
            mov     p2,crcmsb   ;p2=00 read crcmsb register
            mov     p3,crclsb   ;p3=00 read crclsb register
            mov     crclsb,#075h ;check crc register operation
                                ;data in = 75 result = E7C1
            mov     crclsb,#08Ah ;data in = 8A result = 37A7
            mov     crclsb,#00Bh ;data in = 0B result = 7D37
            mov     crclsb,#075h ;data in = 75 result = 31FD
            mov     crclsb,#0C7h ;data in = C7 result = 13B1
            mov     crclsb,#0AAh ;data in = AA result = 0B53
            mov     crclsb,#075h ;data in = 75 result = DA8A
            mov     crclsb,#0C7h ;data in = C7 result = 351A
            mov     crclsb,#055h ;data in = 55 result = F474
            mov     crclsb,#043h ;data in = 43 result = D6B5

            nop                 ;delay after last write and before first read
                                ;let CRC finish
            mov     p0,crcmsb   ;p0=D6 read CRCMSB register
            mov     p1,crclsb   ;p1=B5 read CRCLSB register
            mov     crclsb,crclsb ;clear CRC, data in = B5 result = 00D6
            nop                 ;need delay
            mov     crclsb,crclsb ;cleared, data in = D6 result = 0000

            nop
            mov     p2,crcmsb   ;p1=00 read crcmsb resgister
            mov     p3,crclsb   ;p1=00 read crclsb register
end_loop:
            sjmp    $
            end
```

As mentioned, the CRC–16 function is optionally available to the application software. This is available regardless of whether the automatic power–on CRC is used. Although a CRC could be computed completely in software, it would take much longer than using the DS5001 facility. Using the CRC–16 hardware, the DS5001 series can perform a CRC–16 on 64K bytes of memory in approximately 500 ms. The CRC–16 logic resides behind the two SFRs mentioned above. These display the current CRC result and also serve as the input locations. The software must sequentially write the memory values into the CRC LSB at location 0C2h. After a delay of one instruction cycle, the 16–bit result will be available at 0C3h and 0C2h. The CRC–16 is a superior method of checking the file validity compared to a checksum. Using the DS5001hardware, it can be computed quickly. When using the CRC–16 hardware as part of an application, the existing CRC should first be cleared. This is done by writing the CRC back on itself. This process makes the CRC–16 result equal to 0000h. The LSB is written back twice with a delay in between for computation. A code example is shown in Figure 8–3. This displays the result of the CRC–16 on ports 0 and 1.

SECTION 9: FIRMWARE SECURITY

One of the most unique features of the Soft Micro is its firmware security. The family far surpasses the standard offering of ROM based microcontrollers in keeping system attackers or competitors from viewing the contents of memory. In a standard EPROM based microcontroller, a knowledgeable attacker can disable the EPROM security bit and have access to the entire memory contents. The Soft Micro's improved security makes it a natural choice for systems with high security requirements such as financial transaction terminals. However, the firmware security can also be employed to keep competitors from copying proprietary algorithms. Allowing access to these algorithms can create an instant competitor. This section describes the security features and their application. Also included are guidelines to using microcontroller security within the framework of total system security.

As with memory map control, there are variations between the different Soft Micro versions. The original DS5000 has a high level of firmware security that was the best available when it was introduced. Since then, the DS5002 has added several distinct improvements. Note that the DS5001 has only minimal security and should only be applied when other physical security is used or when security is not needed. The table below provides a brief summary of the versions and their security features. A detailed description of each feature follows. In the description, elements that are unique to a particular Soft Micro version have that version underlined.

FEATURE	DS5001	DS5000	DS5002
Security Lock	Yes	Yes	Yes
RAM memory	Yes	Yes	Yes
Encrypted memory	None	Yes, user must enable	Yes
Encryption algorithm	None	Version 2, circa 1989	Version 3, circa 1991
Encryption Key	None	48–bits	64–bits
Encryption Key Selection	None	User selected	True random number
Encryption Keys loaded	N/A	When user selects	Automatic, any new load, dump
Dummy bus access	None	Yes, when encrypted	Yes
On–chip Vector RAM	None	Yes, when encrypted	Yes
Self–Destruct Input	None	None	Yes
Die Top Coating	None	None	Yes, Second Quarter 1993
Random Number Generator	Yes	None	Yes

SECURITY OVERVIEW

A system might require these security features if it dispenses services on a pay per service basis. Electronically bypassing the security would allow the dispensing of the service for free, resulting in lost revenue to the system owner. Another common application is the transmission of secret information. The user's algorithm and key data could be observed in a unsecured system, resulting in a break in the secure transmission. The Soft Micro family is designed to protect the contents of memory from being viewed. This is done with a combination of circuit techniques and physical security. The combination is a formidable defense. Regardless of the application, the secure microcontroller protects the contents of memory from tampering and observation. This preserves secret information, access to services, critical algorithms etc. The security features of the Soft Micro include physical security against probe, memory security through cryptographic scrambling, and memory bus security preventing analysis of the CPU's operation. The features mentioned above and described below protect the application code and data.

SECURITY LOCK

Ordinarily, the easiest way to dump (view) the memory contents of a Soft Micro is using the Bootstrap Loader. On request, the Loader will transfer the contents of memory to a host PC. This is prevented by the Security Lock. The lock is the minimal security feature, available even in the DS5001. Once set, the Security Lock prevents the Loader from gaining access to memory. In fact, no Loader commands (except Unlock) will work while the Lock is set. The Security Lock is similar in function to an EPROM security bit on a single chip microcontroller. It prevents a programmer from reading the memory. In addition, the Security Lock prevents the micro from executing code on the Expanded bus of Ports 0 and 2. Thus an attacker can not add a memory and use MOVC instructions that would force the micro to read out the contents of protected memory. However, the Soft Micro Security Lock does provide one important difference from EPROM security bits. When the Security Lock is cleared, it destroys the RAM memory contents. If a knowledgeable user were to physically erase the security bit in an EPROM–based micro, the memory contents would remain to be read. The Security Lock consists of a multiple bit latch distributed throughout the micro chip with circuits that collapse the lock in the event of tampering. Clearing the lock results in an unstoppable destructive process that acts differently for each version as described below.

In a DS5001 clearing the lock causes the loader to manually write over the first 32K bytes of NVRAM with zeros. Thus the contents of memory would be erased. This is obviously a low level of security but would deter casual inspection. In a DS5000 or DS5002, clearing the lock causes an instantaneous erasure of the Encryption Key and Vector RAM. This action is unpreventable once the lock is cleared and happens independent of V_{CC} or crystal. Once the erasure has occurred, a DS5000, assumes a non–secure (brand–new) state. In a DS5002, the Loader proceeds to load a new Encryption Key once the erasure has occurred. In both, the Bootstrap Loader will then proceed to overwrite the first 32K bytes of RAM if power is available and the crystal is still present. This last action is for thoroughness. In systems that really require security, the Lock should be combined with Memory Encryption (discussed below). Thus the instantaneous erasure of the Encryption Key renders the contents of memory useless since it can no longer be properly deciphered.

The Security Lock is set via the Bootstrap Loader using the "Z" command. Once issued, the Loader will continue to communicate with a user but will not perform other commands. The Loader will respond with an error message in the event that further commands are issued. While the Lock is set, the Loader has no access to the Byte–wide bus memory. The Security Lock can be cleared using the "U" command. Issuing this command to a locked part results in the destructive process described above. No confirmation is requested. The status of the Security Lock can be read by application software at MCON.0. This bit is only a status flag and can not be affected by the software.

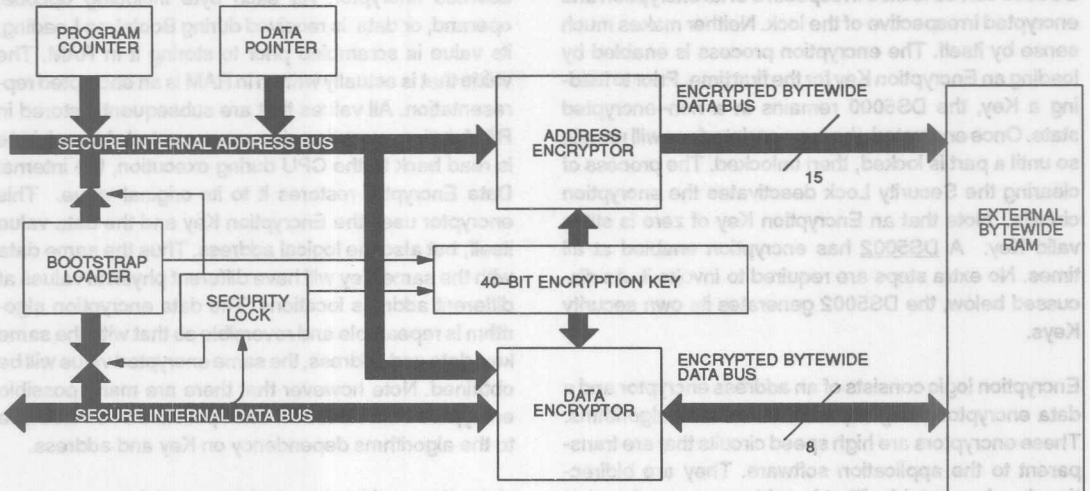
RAM Memory

NVRAM provides a useful way to store program and data. The contents can be retained for a long period, but can be changed when desired. This attribute is important when considering security. No matter what probing techniques are used on a ROM, the contents will remain for viewing. With resources and patience, a determined attacker will obtain the contents of a ROM based product. NVRAM can be destroyed on demand. The user's physical security must simply remove the power (V_{CC} and V_{BAT}) from a micro chip to eliminate the memory contents. Thus while NVRAM provides flexibility, it also provides security. Enough physical security can be combined with even a DS5001 to provide a very secure system. The DS5002 even provides a direct facility to destroy memory discussed below.

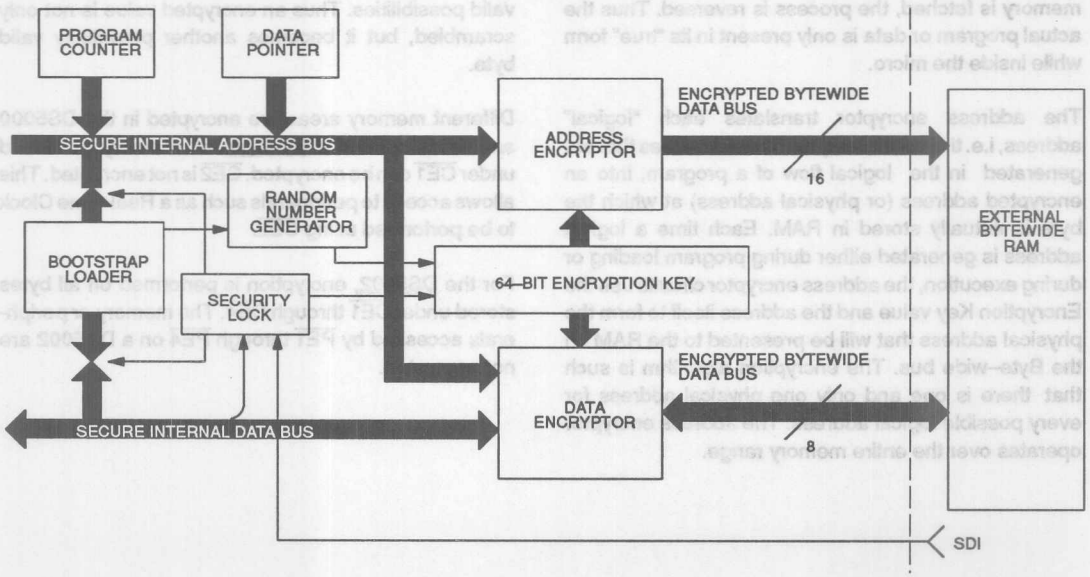
Encrypted Memory

The heart of Soft Micro security is the memory encryption function. Since the NVRAM is visible, the memory contents and memory bus are encrypted. That is, in real time, the addresses and data moving between the RAM and the micro are scrambled by on–chip encryption circuits. Thus an attacker that observes the RAM contents or memory bus will see unintelligible addresses and data. Figure 9–1 shows the conceptual diagram of the memory encryptor for a DS5000 series device. Figure 9–2 shows the encryptor for a DS5002.

DS5000 SOFTWARE ENCRYPTION BLOCK DIAGRAM Figure 9–1



DS5002 SOFTWARE ENCRYPTION BLOCK DIAGRAM Figure 9–2



In a DS5000, the encryption feature is optional. A DS5000 can be locked irrespective of its encryption and encrypted irrespective of the lock. Neither makes much sense by itself. The encryption process is enabled by loading an Encryption Key for the first time. Prior to loading a Key, the DS5000 remains in a non–encrypted state. Once encrypted, the memory interface will remain so until a part is locked, then unlocked. The process of clearing the Security Lock deactivates the encryption circuits. Note that an Encryption Key of zero is still a valid Key. A DS5002 has encryption enabled at all times. No extra steps are required to invoke it. As discussed below, the DS5002 generates its own security Keys.

Encryption logic consists of an address encryptor and a data encryptor using separate but related algorithms. These encryptors are high speed circuits that are transparent to the application software. They are bidirectional and repeatable. That is, addresses and data that are scrambled prior to writing to RAM will be correctly unscrambled when reading in reverse. Each encryptor operates with its own algorithm but both are dependent on the Encryption Key. Encryptors operate while programs are being loaded so that the memory contents are stored in its scrambled form. When program memory is fetched, the process is reversed. Thus the actual program or data is only present in its "true" form while inside the micro.

The address encryptor translates each "logical" address, i.e. the normal sequence of addresses that are generated in the logical flow of a program, into an encrypted address (or physical address) at which the byte is actually stored in RAM. Each time a logical address is generated either during program loading or during execution, the address encryptor circuits use the Encryption Key value and the address itself to form the physical address that will be presented to the RAM on the Byte–wide bus. The encryption algorithm is such that there is one and only one physical address for every possible logical address. The address encryptor operates over the entire memory range.

The Data Encryptor operates in a similar manner to the address encryptor. As each byte including opcode, operand, or data is received during Bootstrap Loading, its value is scrambled prior to storing it in RAM. The value that is actually written in RAM is an encrypted representation. All values that are subsequently stored in RAM during execution also are encrypted. As each byte is read back to the CPU during execution, the internal Data Encryptor restores it to its original value. This encryptor uses the Encryption Key and the data value itself, but also the logical address. Thus the same data with the same Key will have different physical values at different address locations. The data encryption algorithm is repeatable and reversible so that with the same key, data and address, the same encrypted value will be obtained. Note however that there are many possible encrypted data values for each possible true value due to the algorithms dependency on Key and address.

Using the combination of address and data encryption, the normal flow of program code is unintelligible in the NVRAM. What had been a sequential flow of addresses is now apparently random. The values stored in each memory location appear to have no relation to the original data. Another factor that makes analysis more difficult is that all 256 possible values in each memory are valid possibilities. Thus an encrypted value is not only scrambled, but it becomes another potentially valid byte.

Different memory areas are encrypted in the DS5000 and DS5002. For a DS5000, all memory accessed under $\overline{CE1}$ can be encrypted. $\overline{CE2}$ is not encrypted. This allows access to peripherals such as a Real–time Clock to be performed using $\overline{CE2}$.

For the DS5002, encryption is performed on all bytes stored under $\overline{CE1}$ through $\overline{CE4}$. The memory or peripherals accessed by $\overline{PE1}$ through $\overline{PE4}$ on a DS5002 are not encrypted.

Encryption Algorithm

Soft Micros use a proprietary algorithm to encrypt memory. The DS5000 and DS5002 use different encryption algorithms. They are the result of improvements made over time in the proprietary encryptor circuits. The original DS5000 (circa 1988) has the first version of encryptor. This was soon improved with a second version encryptor in 1989, and remains in production today. A substantial improvement was made in the DS5002, which uses a wider Key and a more non–linear algorithm. The DS5002 memory encryptor uses elements of the DES (Data Encryption Standard) although not the entire algorithm. Full DES is impractical as memory encryption must be performed in real–time on a one–to–one substitution and not a block cypher basis. The encryption algorithm is supported by the fact that both address and data are encrypted, the algorithm and key are both secret, the most critical data can be stored on chip in vector RAM (discussed below), and the bus activity is scrambled using dummy access (discussed below). For this reason, a security analysis of the DS5002 is not simply a mathematical treatment of the encryption algorithm.

Encryption Key

The DS5000 uses a 40–bit Encryption Key that is stored on–chip. As mentioned above, the Key is the basis of the encryption algorithm. The resulting physical addresses and data are dependent on this value. Tampering with or unlocking the micro will cause the Key to be instantaneously destroyed. If the memory contents are encrypted, they become useless without this Key. A user selects the 40–bit Key and loads it via the Bootstrap Loader. Selecting this Key enables the encryption feature. The DS5002 uses a 64–bit Key. It is similarly stored on–chip in tamper resistant circuits. In much the same way, this Key is the basis for the physical values that are presented on the bus. Using a wider Key gives the encryption more complexity and more permutations that must be analyzed by an attacker. Apart from the width of the Key and complexity of the encryptor, the principal differences between the DS5000 and DS5002 are discussed below under Key Selection and Loading.

Encryption Key Selection and Loading

One of the significant differences between DS5000 and DS5002 lies in Encryption Key Management. In the case of a DS5000, the user must select a 40–bit Key during program loading. This Key must be selected prior to loading the micro, as the memory will be encrypted as it is loaded. The Key selection process must be protected since an attacker that learns the Key can reproduce the user's code. This would be done by loading the correct Key in an unlocked DS5000FP, attaching the encrypted memory chip, and dumping the code using the Bootstrap Loader.

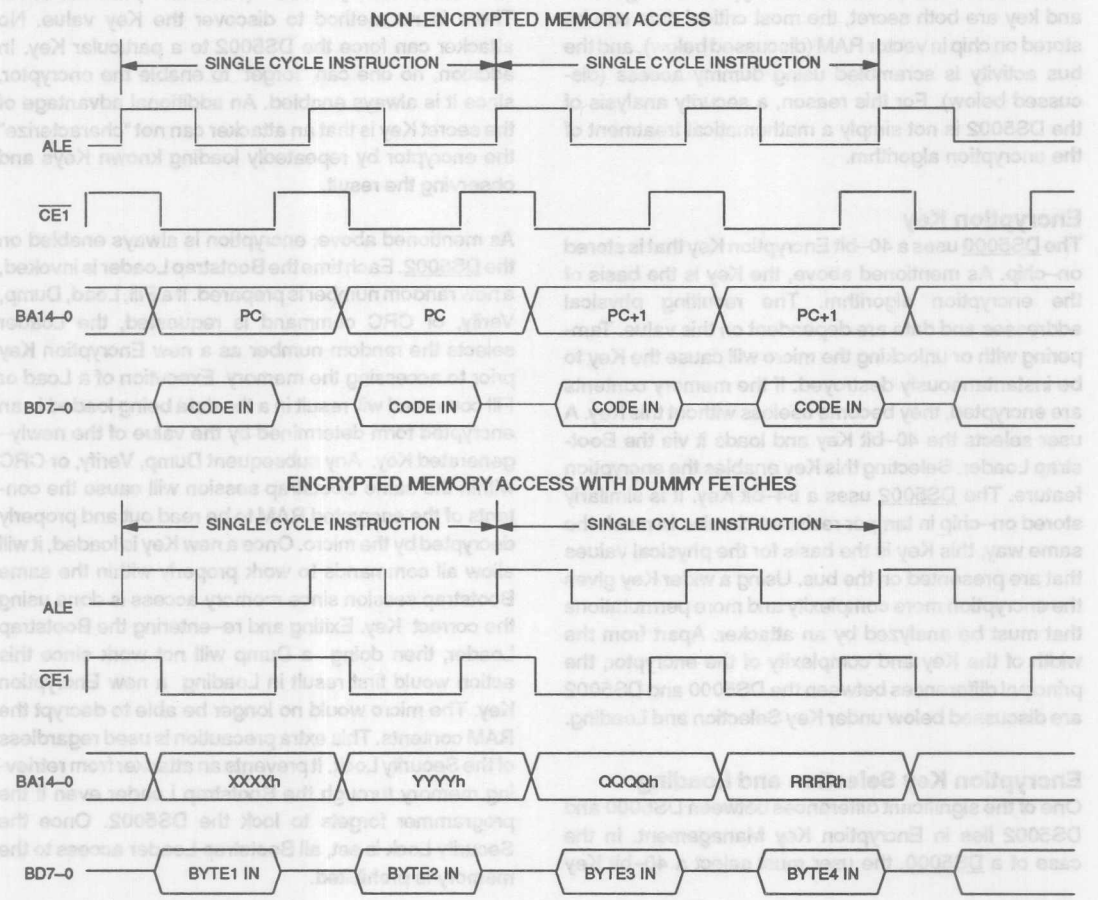
The DS5002 provides an improved Key management system. The micro chooses its own 64–bit Encryption Key from a number that is internally generated and secret. The Keys come from a true hardware random number generator. It is based on frequency differences between two on–chip ring oscillators and the user's crystal. At any time, it is unlikely that any two DS5002s have the same key with 2^{64} ($1.84 * 10^{19}$) combinations. There is no method to discover the Key value. No attacker can force the DS5002 to a particular Key. In addition, no one can "forget" to enable the encryptor, since it is always enabled. An additional advantage of the secret Key is that an attacker can not "characterize" the encryptor by repeatedly loading known Keys and observing the result.

As mentioned above, encryption is always enabled on the DS5002. Each time the Bootstrap Loader is invoked, a new random number is prepared. If a Fill, Load, Dump, Verify, or CRC command is requested, the Loader selects the random number as a new Encryption Key prior to accessing the memory. Execution of a Load or Fill command will result in a the data being loaded in an encrypted form determined by the value of the newly–generated Key. Any subsequent Dump, Verify, or CRC within the same Bootstrap session will cause the contents of the encrypted RAM to be read out and properly decrypted by the micro. Once a new Key is loaded, it will allow all commands to work properly within the same Bootstrap session since memory access is done using the correct Key. Exiting and re–entering the Bootstrap Loader, then doing a Dump will not work since this action would first result in Loading a new Encryption Key. The micro would no longer be able to decrypt the RAM contents. This extra precaution is used regardless of the Security Lock. It prevents an attacker from retrieving memory through the Bootstrap Loader even if the programmer forgets to lock the DS5002. Once the Security Lock is set, all Bootstrap Loader access to the memory is prohibited.

Dummy Bus Access

The Soft Micro security makes the memory contents obscure through encryption. However, the Soft Micro takes additional steps to prevent analysis of the bus activity by an attacker that is knowledgeable about the 8051. To further obscure bus activity, both the DS5000 and DS5002 insert dummy memory access when possible. In examining the 8051 architecture, there are typically two memory accesses per instruction cycle. In the standard 8051, these are identical. Nothing is actually done with the second program fetch. In the Soft Micro, a pseudo–random address is generated for the dummy cycle and this random memory address is actually fetched. The dummy data is not actually used inside of the micro. The order of the real and dummy accesses are switched according to a pseudo–random process. This is repeatable so that the execution always appears the same. During these pseudo–random cycles, the RAM is to all appearance read. Thus by repeatedly switching between real and dummy access, it is impossible to distinguish a dummy cycle from a real one. In analyzing bus activity, a large percentage of the memory fetches will be garbage that has no meaning. The dummy accesses are always performed on a DS5002, but are only used on a DS5000 when encryption is enabled. Naturally, dummy accesses are always read operations since the dummy address might contain valid data.

DUMMY BUS ACCESS TIMING Figure 9–3



Either XXXX or YYYY is real but encrpted, the other is pseudo–random.
Either QQQQ or RRRR is real but encrypted, the other is pseudo–random.
Either Byte1 or Byte2 is used, the other is a dummy fetch and is not used. Both are encrypted.
Either Byte3 or Byte4 is used, the other is a dummy fetch and is not used. Both are encrypted.

On–chip Vector RAM

A 48–byte RAM area is incorporated inside the DS5000 and DS5002 micro chips. This area maps to the first 48 locations of program memory to store reset and interrupt vectors. Any other data stored in the first 48 locations will be contained in this Vector RAM. The principal reason for the Vector RAM is that the reset and interrupt vectors are known logical addresses in the 8051 family. Thus an attacker could force a reset or interrupt and discover the encrypted address generated by the Soft Micro. By storing these Vectors in on–chip RAM, it is impossible to observe such relationships. Although it is very unlikely that an application program could be deciphered by observing the vector addresses, the Vector RAM eliminated this possibility. Note that the dummy accesses discussed above also occur while the Vector area is being accessed.

The Vector RAM is automatically loaded with the reset and interrupt vectors during Bootstrap Loading. This feature is transparent to operation and no action is required to use it. However, considering the Vector area feature can improve overall system security. As mentioned above, the Vector RAM is instantaneously destroyed in the event of an unlock (also a self–destruct on DS5002). Since it is hidden and subject to destruction, the 48–bytes are the most secure memory in a system. Thus the most critical constants can also be stored there. This is an ideal location for storing DES keys for applications involving data encryption such as electronic funds transfer.

The Vector RAM is always used on a DS5002. The data stored between logical location 00h and 30h will be loaded into and executed for the Vector RAM. This data will not be duplicated in NVRAM accessed by the Byte–wide bus. The operation of DS5000 Vector RAM is the same, but only when the encryption feature is enabled. When a DS5000 has not had an Encryption Key loaded, the Vector RAM is left unused.

Self–Destruct Input

The Self–Destruct Input (SDI) is an active high input pin that is used to clear the security lock on a DS5002 in response to an external event. The SDI is intended to be used with external tamper detection circuitry. It can be activated by an active high signal with or without operating power applied to the V_{CCI} pin. Activation of the SDI pin instantaneously clears the Security Lock causing the sequence of events described above. In addition, power is momentarily removed from all Byte–wide bus interface signals including the V_{CCO} pin, resulting in loss of data by the external RAM. Address and data lines are also pulled low to remove any excess charge that could help retain data in that RAM. The SDI pin is deglitched so that a 2 µs pulse is required to activate it. However, this pin is sensitive so it should be grounded if not used. It is only available on the DS5002FP or DS2252 products.

Die Top Coating

Available as of Second Quarter 1993. The DS5002 is provided with a special top–layer coating that is designed to prevent a micro probe attack. The coating is implemented with a second layer of metal on the microcontroller die. This metal will result in a short circuit of critical functions if probing is attempted. The probing action destroys the data that is secret. Also, security circuits and Vector RAM derive their power from this screen. Therefore they will be de–powered if the top coating is removed, also destroying the secret data. In this event, any critical data stored on–chip will be destroyed and off–chip data is rendered useless.

Random Number Generator

As mentioned above, the DS5002 incorporates a hardware random number generator used by the Bootstrap Loader to generate Encryption Keys. The Random Number Generator is not a security circuit perse, but it is available to the application and can be used to improve the overall system security. Random numbers have numerous applications with respect to security. For example, to prevent an attacker from developing a histogram of code execution, the Random Number Generator could be used to decide how long to spend on particular activities. The random number is created 8–bits at a time. They are obtained by the application code at SFR location 0CFh. The random number takes 160 µs to develop. Once a byte is created, another will not be generated unless the previous value is read. After the random number is read, another will be available approximately 160 µs later. The RNR bit (RPCTL.7; 0D8h) will be set to a logic 1 each time a new number is available. If the random number is read prior to RNR being set, the value will be 00.

Security Summary by Part

The preceding information outlined each of the security features. Their inclusion in various parts is shown in the table at the beginning of this chapter. For completeness, the following is a summary description of security features for each part in the Soft Micro family.

DS5000

The DS5000 is the second generation of a microcontroller with security. The first is an earlier version of DS5000 circa 1988, now obsolete. The DS5000 incorporates a combination of real–time memory encryption and Security Lock. The memory encryption is optional however. To invoke the encryption, the user must select a 48–bit Encryption Key using the Bootstrap Loader. A user then loads the memory which will be automatically encrypted using this Key. After the memory is loaded and verified, the DS5000 can be locked. Locking the micro prevents an attacker from using the Bootstrap Loader to decrypt and dump the memory contents. Unlocking the DS5000 destroys the Encryption Key and Vector RAM. Vector RAM is 48–bytes of secret storage on–chip. It is used to hold reset and interrupt vectors as well as any application values than must be hidden. In addition to encrypting the memory, the DS5000 generates dummy bus cycles to obscure the actual program flow. Dummy cycles appear to be actual memory fetches but are not actually used inside the micro. Also fundamental to the security of a DS5000 is its basis on RAM. This allows all security features to be changed frequently. The strategy is that an attacker must spend a long time breaking into the DS5000, but the user can simply change system security at any time. Thus any information that is successfully stolen has a very limited lifetime. This overview applies to the DS5000FP, DS5000(T), and DS2250(T) products.

DS5001

The DS5001 is a newer product than the DS5000, but has less security. It is useful in systems that need a large memory, but that provide sufficient physical security for all needs. The DS5001 incorporates a Security Lock. This is used to prevent the Bootstrap Loader from dumping memory. Once locked, the Bootstrap Loader can not access the memory. Unlocking the DS5001 causes the Bootstrap Loader to write over the NVRAM. The RAM nature of he DS5001 product allows a user to vary security frequently and to manually destroy it if necessary. This overview applies to the DS5001FP and DS2251(T) products.

DS5002

The DS5002 adopts the memory and I/O improvements of the DS5001 and improves on the security of the DS5000. It is a high security version of the DS5001. This device is intended for maximum security and has numerous improvements to the DS5000. The security is always enabled on a DS5002. Thus an attacker can not characterize the security and the user can not forget to enable the security. The DS5002 follows a similar scheme of memory encryption and Security Lock. The DS5002 encryptor is a superior algorithm using a 64–bit Encryption Key. In addition, the Key is managed by the DS5002. Using the Bootstrap Loader, each part selects a random number for its 64–bit Key and installs it prior to loading memory. Leaving and re–entering the Bootstrap loader causes the DS5002 to select a new number as a potential Key. Any subsequent memory access with the Loader causes the new Key to be installed. Like the DS5000, the DS5002 also uses dummy bus access and Vector RAM to further hide memory bus activity. The Security Lock of a DS5002 is similar in nature to the DS5000. Once locked, the DS5002 Bootstrap Loader does not have access to memory. Unlocking the DS5002 destroys the Encryption Key and Vector RAM. The NVRAM accessed by the Byte–wide bus is also manually erased under Bootstrap Loader control. The DS5002 provides an external method to clear the Security Lock using its Self–Destruct Input (SDI). This causes the erasure of the Key and Vector RAM and also removes power from the NVRAM. The DS5002 provides physical security for the microcontroller die with a metal covering to prevent microprobe. This overview applies to the DS5002FP and DS2252(T) products.

APPLICATION: SYSTEM SECURITY

The Soft Micro family has been used for numerous applications requiring security. Different levels of security are required depending on the sensitivity of the application and the value of the protected information. As mentioned above, the goal of the microcontroller security is to make stealing the protected information more difficult than the information is worth. This task actually has two pieces. First, the Soft micro makes attack difficult. This is combined with the user's physical security to make information retrieval difficult. The second part is to make the protected information less valuable. To this end, the NVRAM nature allows a user to frequently alter the firmware based security aspects of the system. Thus if the critical information changes before the security can be broken, the information that is actually retrieved will be worthless.

To assess the security of a system, the total implementation must be examined. The DS5000 or DS5002 provides a high level of security, but the user's firmware can accidentally defeat some features. Below are a sampling of implementation issues that will make the DS5000 or DS5002 more difficult to crack. There are also suggestions on making a system more secure using external circuits.

Avoid Clear Text

The encryption algorithms used by DS5000 or DS5002 are generally adequate to prevent analysis when combined with well developed code. However, the encryption is defeated to some extent if the user stores text that appears on a display in encrypted form. This gives the pirate a starting point to look for the clear text in encrypted storage and analyze the encryption algorithm. The "data answer" is already known. If clear text is required, then preferably store it in nonencrypted memory. If this is impractical, then disperse it so that it is hard to find. Avoid at all costs reading the clear text from memory then immediately displaying it. This is a sure means to identify the encrypted values of the text for the attacker.

Avoid CRC or Checksum

Running a checksum on power up provides the pirate with a sequential listing of the addresses in encrypted form. Therefore the attacker has a great advantage in deciphering the Address Encryptor. Preferably avoid a checksum. If one is needed, then check the minimum amount of memory and perform the check in non–sequential fashion.

Avoid Long Straight Runs of Code

A common coding practice is to run numerous sequential operations. This is common knowledge and should be avoided. The pirate can use this in the same way as a checksum process. It provides a sequential listing of encrypted addresses and assists with analysis of the address encryption.

Use Jumps

To address the prior problem, jumps are advised. These can be jumps for no reason other than to space out straight runs of code. However, using jumps also provides several other techniques to make bus analysis more difficult. As an example, the code can jump into Vector RAM. While in this area, dummy access will occur on the bus.

Use Random values

The Random Number Generator of the DS5002 can be used to make a pirate's task more difficult. When time is available, the software should perform random actions at random time intervals. As an example, the Random Number Generator can be used to select a timer interrupt value. Thus the micro will be interrupted at random intervals making characterization very difficult. Software can elect to remain in Vector RAM for a random period of time. Also as discussed above, the micro generates dummy RAM reads when possible. However, it can not generate dummy writes. However the user's code can. Random numbers can be written to address that are known to be unused. If this is done while the micro is visibly performing a meaningful task, it will make analysis very difficult.

Vector RAM

As mentioned above, the Vector RAM can be used for many things beside vectors. This is the most secure storage in the system. It resides on–chip behind tamper protection. Thus it is useful for storing the most sensitive data. Thus even an attacker could break the encryption, this information would still be secret. For EFT or similar applications, this is a good location for the storage of DES keys. Since DES is a public algorithm, the real protection is keeping the DES key secret. As this is only 8 bytes, it fits well within the Vector RAM.

Change Code

Perhaps most importantly, the user should reprogram portions of the Soft Micro that deal with security. For example, if the micro is performing DES, the user can change DES keys. Any security system can be broken with enough time and resources. By altering the security features, this threat can be minimized.

External Circuits

A variety of external circuits can support secure operation. For example, the DS2400 is a unique 48–bit Silicon Serial Number. If it is installed with the micro, it can be read when the system is first powered up, then stored inside the Soft Micro. This serializes the system. If the software ever finds a different serial number (or missing number) from the stored one, it can refuse to work. This would mean that the micro had been moved.

Tamper Protection

Using a variety of tamper sensors in conjunction with the DS5002 makes the system very difficult to crack. These circuits vary from simple switches to light, temperature, pressure, or oxygen sensors. When the physical security is violated, the SDI pin is activated and the memory contents are destroyed.

SECTION 10: RESET CONDITIONS

Reset Sources

The Soft Microcontroller is designed to provide proper reset operation with a minimum of external circuitry. In fact, for may applications, external reset circuitry is not required. The possible sources of reset are as follows:

a) Power On (operating voltage applied to V_{CC})

b) No V_{LI} Power On

c) External RST pin

d) Watchdog Timeout

Certain actions are taken in all cases where a reset has been issued. Whenever any type of reset is executed, the ALE and PSEN quasi-bidirectional pins are configured as inputs. In addition, an internal reset line (IRST) is active continuously until the condition which is causing the reset has been removed. IRST will then go inactive and execution of the application program will begin. Special Function Registers are initialized during reset as shown in Table 10–1.

Figure 10–1 is a summary of the bits that indicate the source of the most recent reset. Operational details which are unique to the different sources of reset are discussed below:

RESET STATUS BITS Figure 10–1

PCON.6:	**POR**
"Power On Reset":	Indicates that the previous reset was initiated during a Power On.
Initialization:	Cleared to a 0 whenever a Power On Reset occurs; remains unchanged on other types of resets. Must be set to a 1 by software.
Read Access:	Can be read normally anytime.
Write Access:	Can be written only by using the Timed Access register.
PCON.4:	**WTR**
"Watchdog Timer Reset":	Set to a 1 when a timeout condition of the Watchdog Timer occurs. Cleared to a 0 immediately following a read operation.
Initialization:	Set to a 1 on a Watchdog Timeout Reset. Remains unchanged on any other type of reset.
Read Access:	Read normally anytime.
Write Access:	Not writable.
PCON.2:	**EWT**
"Enable Watchdog Timer":	The Watchdog Timer is enabled if EWT is set to a 1 and is disabled if EWT is cleared to a 0. This is not normally considered a status bit but is convenient for detecting a No V_{LI} reset condition.
Initialization:	Cleared to a 0 on a No-V_{LI} Power On Reset. Remains unchanged during other types of reset.
Read Access:	May be read normally anytime.
Write Access:	Writable only by using the Timer Access register.

SPECIAL FUNCTION REGISTER RESET STATES Table 10–1

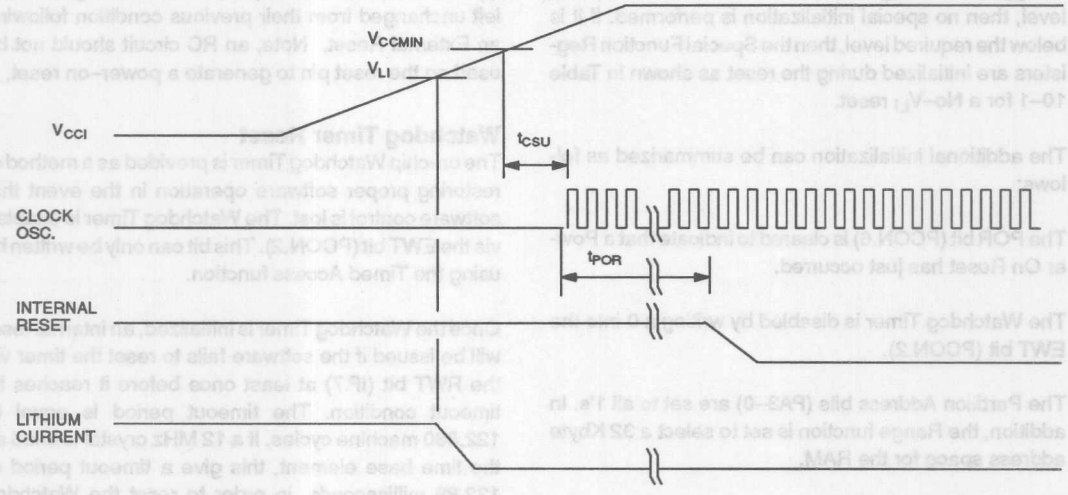
REGISTER	LOCATION	RESET CONDITION	RESET TYPE
PC	N/A	0000h	All
ACC	E0h	00h	All
B	F0h	00h	All
PSW	D0h	00h	All
SP	81h	07h	All
DPTR	83h, 82h	0000h	All
P0–P3	80h, 90h, A0h, B0h	FFh	All
IP	B8h	0XX00000b	All
IE	A8h	0XX00000b	All
TMOD	89h	00h	All
TCON	88h	00h	All
TH0	8Ch	00h	All
TL0	8Ah	00h	All
TH1	8Dh	00h	All
TL1	8Bh	00h	All
SCON	98h	00h	All
SBUF	99h	XXXXXXXXb	All
PCON	87h	0UUU0U00b 00000U00b 00000000b 0U010U00b	External reset Power on reset No V_{LI} reset Watchdog Timer
MCON (DS5000)	C6h	UUUUUU0Ub UUUUUU0Ub 11111000b UUUUUU0Ub	External reset Power on reset No V_{LI} reset Watchdog Timer
MCON (DS5001)	C6h	UUUUUU0Ub UUUUUU0Ub 11111000b UUUUUU0Ub	External reset Power on reset No V_{LI} reset Watchdog Timer
Encryption Key (DS5000)	N/A	UUh UUh UUh UUh UUh UUh UUh UUh UUh UUh Disabled UUh UUh UUh UUh UUh	External reset Power on reset No V_{LI} reset Watchdog Timer
RPCTL (DS5001)	D8h	0X00000Ub 0X00000Ub 0X000000b 0X00000Ub	External reset Power on reset No V_{LI} reset Watchdog Timer
Status (DS5001)	DAh	00h	All
RNR (DS5001)	CFh	XXh	All
CRC (DS5001)	C1h	UUUUXXUUb UUUUXXUUb 0000XX00b UUUUXXUUb	External reset Power on reset No V_{LI} reset Watchdog Timer
CRC High (DS5001)	C3h	00h	All
CRC Low (DS5001)	C2h	00h	All

NOTES:
X indicates a bit that is indeterminate on a reset.
U indicates a bit that is unchanged from its previous state on a reset.

Power On Reset

The Soft Micro provides an internal Power On Reset capability which requires no external components. When voltage is applied to the V_{CC} pin from a power off condition, the Soft Micro automatically performs an internal reset sequence to prepare the processor for execution of the application software. The traditional capacitor reset circuit should not be used. Figure 10–2 illustrates the timing associated with the Power On Reset cycle.

POWER ON RESET TIMING Figure 10–2



This cycle begins with Power On reset delay time. This is generated by the internal control circuitry to allow the internal clock oscillator to start up from its halted state that is in effect when V_{CC} is below V_{CCmin}. The period t_{CSU} is a mechanical startup time that is dependent on the individual crystal. The delay shown as t_{POR} in the figure is generated by internal circuitry which counts a total of 21,504 (1.792 ms @ 12 MHz) clock oscillator periods before it allows the internal reset line to be released. The purpose of this delay is to allow time for the clock frequency to stabilize.

The Power On Reset delay is not the total amount of time which must pass before execution can begin in the application from the initial application of V_{CC} voltage. First the power supply slew rate is required for V_{CC} to rise from 0V to the V_{CCmin} threshold shown in Figure 10–2. Next, operation with a crystal is partly mechanical and some time is required to get the mass of the crystal into vibrational motion. The user should consult the crystal vendor for a start-up time specification.

When a Power On Reset cycle is in progress, the external RST pin has no effect on internal operation. Once control of the processor is transferred to the user's program, a hardware reset may be issued externally via the RST pin.

A Power On Reset causes special initialization to be performed on the Special Function Registers as shown in Table 10–1.

The distinguishing action taken during a Power On Reset is that the $\overline{POR}$ bit is cleared in order to indicate that a Power On Reset has just occurred. All other control bits which are initialized according to the type of reset are left unchanged from their previous condition.

No-V_{LI} Power On Reset

During a Power On Reset cycle, a test is automatically performed by the internal control circuitry to measure the voltage of the lithium power source. This test determines whether or not the current lithium voltage (V_{LI}) is above the minimum level required (V_{LImin}) to insure that the nonvolatile areas can be maintained in the absence of V_{CC}. If the voltage is found to be above the required level, then no special initialization is performed. If it is below the required level, then the Special Function Registers are initialized during the reset as shown in Table 10–1 for a No–V_{LI} reset.

The additional initialization can be summarized as follows:

The POR bit (PCON.6) is cleared to indicate that a Power On Reset has just occurred.

The Watchdog Timer is disabled by writing a 0 into the EWT bit (PCON.2).

The Partition Address bits (PA3–0) are set to all 1's. In addition, the Range function is set to select a 32 Kbyte address space for the RAM.

On a DS5000, the Encryption Key Word and software encryption operation are disabled.

Finally, the Security Lock bit is cleared to 0.

External Reset

For applications which require an external reset capability, a reset pin (RST) is provided with a Schmitt Trigger input. This input may be used to force a reset condition any time when the micro is executing the application program or when it is in either the Idle or Stop modes. Reset is initiated by holding the RST pin active (high) for a minimum time of two machine cycles (24 clock oscillator periods). If the reset was initiated from Stop mode, the rising edge will result in an internally-generated Power On Reset time (t_{POR}) which is required for the oscillator to start and for the clock frequency to stabilize.

All of the control bits that are initialized according to the type of reset within the Special Function registers are left unchanged from their previous condition following an External Reset. Note, an RC circuit should not be used on the reset pin to generate a power–on reset.

Watchdog Timer Reset

The on-chip Watchdog Timer is provided as a method of restoring proper software operation in the event that software control is lost. The Watchdog Timer is enabled via the EWT bit (PCON.2). This bit can only be written by using the Timed Access function.

Once the Watchdog Timer is initialized, an internal reset will be issued if the software fails to reset the timer via the RWT bit (IP.7) at least once before it reaches its timeout condition. The timeout period is equal to 122,880 machine cycles. If a 12 MHz crystal is used as the time base element, this give a timeout period of 122.88 milliseconds. In order to reset the Watchdog Timer in the application software, the RWT bit must be written with a 1 using the Timed Access procedure. The Watchdog Timer is also reset following any other type of reset.

When a Watchdog Timer reset occurs, special initialization is performed on the Special Function Registers as shown in Table 10–1.

The distinguishing action taken during this type of reset is that the WTR status flag is set to indicated that a Watchdog Timer Reset has just occurred.

APPLICATION: RESET VECTOR

Like the 8051, the Soft Micro will begin execution at address 0000h. This is the Reset Vector. Following this are other vector locations used for interrupts. These begin at address 0003h and are discussed in the User's Guide section covering interrupt operation. Since there are only three memory locations dedicated to the Reset Vector, the user will typically insert a jump statement to a more convenient memory address. This will be the reset routine. It can lie any where in the 64K bytes of program memory addressed by the Soft Micro. A common choice is location 0030h. Thus at location 0000h, the user would use the instruction SJMP 30h. This instruction requires two bytes, so it easily fits in the available space. At the location of the reset routine, the user places instructions that initialize the micro and any external hardware specific to the application. This note describes the operations that are typically done and shows some example code.

At the user's reset routine, the micro must be initialized. The functions that are typically initialized are as follows:

MEMORY	INTERRUPTS	TIMERS/SERIAL	PROTECTION
Partition	Power–fail	Timer setup	Watchdog Timer
Current Memory Map	External	Timer for baud–rates	POR
Data Pointer	Serial Port	Serial Port	
	Timer		

Memory Map

The most critical and most overlooked initialization is that of the memory map. Several of these functions are nonvolatile and are not cleared during a reset. Those that are cleared could leave the micro in an undesirable state. Therefore, the user should either verify the correctness of the memory map or simply set it properly following each reset. An example of how the memory map could be incorrect on reset is as follows.

The user typically sets the Partition, Range, etc., during Bootstrap Loading. In the course of operating however, the user may temporarily move the Partition to alter a lookup table. If while the Partition is moved, a reset should occur, the Partition will remain in the temporary position unless corrected.

In developing the reset routine, the user should carefully note the reset state of each critical bit. For example, when using the ECE2 on a DS5000, note that it is not altered on reset. On a DS5001, the PES bit is cleared on a reset. Thus a DS5000T that is accessing the Real–time Clock when a reset occurs will still be pointing the CE2 space after reset. The DS2251T user that is accessing the RTC when a reset occurs will start in the normal memory configuration.

A code example that initializes the memory map is as follows. It assumes that the DS5000 user requires a Partition of 5800h. A DS5001 using the same code gets a Partition of B000h.

```
MCON    EQU     0C6h
Org     00h

SJMP    Start

Org     30h
Start :

MOV     TA,     #0AAh   ;Timed
MOV     TA,     #55h    ; Access
ORL     MCON,   #02h    ;Set PAA - DS5000 ONLY
MOV     MCON,   #0B8h   ;Set Partition to 5800 on DS5000, B000h on DS5001
MOV     TA,     #0AAh   ;Timed - DS5000 ONLY
MOV     TA,     #55h    ; Access - DS5000 ONLY
ANL     MCON,   #0FDh   ;Clear  PAA - DS5000 ONLY
```

Another common memory requirement is the initialization of the Data Pointer. When using NVRAM to store data, this pointer must be moved to the Partition address (in a partitionable configuration). Thus if the Partition is set to 5800h, the DPTR should be set to 5800h to start. Once data has been saved in NVRAM, the DPTR should be saved in a known, nonvolatile location so that is can be restored on a reset.

Interrupts

After a reset, all interrupts are disabled. Therefore the user must enable individual interrupts that are needed. The global interrupt enable must also be activated. Any interrupt needing a higher priority must be selected as such. The following code example shows the enabling of individual interrupts. A user would combine the appropriate bits as needed by the application. In this application example, the serial port is given a high priority interrupt.

```
ORG     00h
SJMP    Start

Org     30h

Start :

ORL     PCON, #08h      ;Enable Power-fail Warning by setting EPFW
SETB    PS              ;Set Serial Port Interrupt to High Priority
SETB    ES              ;Enable Serial Port Interrupt
SETB    ET1             ;Enable Timer 1 Interrupt
SETB    EX1             ;Enable External Interrupt 1
SETB    ET0             ;Enable Timer 0 Interrupt
SETB    EX0             ;Enable External Interrupt 0

SETB    EA              ; Globally enable interrupts
```

Timers

The Soft Micro disables timer activity (excluding the Watchdog) and serial port communication on a reset. Therefore, each timer must be setup and enabled as part of the reset routine. The serial port mode must also be initialized if used. This is covered in detail in the User's Guide section on Timers and Serial I/O respectively. Shown here is an example of Timer and Serial Port setup. In this example, Timer 0 is set up to generate a 10 ms interrupt. Timer 1 is setup to generate 9600 baud for the serial port. The serial port is set up for asynchronous communication with a PC (mode 1). A crystal frequency of 11.0592 MHz is assumed.

```
        ORG     00h
        SJMP    Start

        Org     30h

        Start :

        SETB    PS              ;Set Serial Port Interrupt to High Priority
        SETB    ES              ;Enable Serial Port Interrupt
        SETB    ET0             ;Enable Timer 0 Interrupt
        MOV     TMOD,   #00100001b      ;Select Timer 1 mode 2 - 8 bit auto-reload,
                                ;       Timer 0 mode 1 - 16 bit manual reload
        MOV     TH1,    #0FDh           ;Setup 9600 baud
        MOV     TL1,    #00h            ; " "
        MOV     TH0,    #0DBh           ;Select a 10 ms count. 9216 counts = 10 ms
        MOV     TL0,    #0FFh           ; 9216d counts = 2400h counts (FFFFh-2400h =
                                        ; DBFFh)
                                        ; Timer 0 ISR must reload DBFFh manually
        MOV     SCON,   #01010011b      ;Select Serial Port mode 1,
                                        ; TXD and RXD interrupts active

        MOV     TCON,   #01010000b      ;Enable the operation of both Timers
        SETB    EA                      ;Globally enable interrupts
```

Protection

The Soft Micro provides protection from transients through a built in power–fail/power–on reset and Watchdog Timer. Each of these functions should be initialized by the user as part of the reset routine. The following code demonstrates the set up for a user that will support the Watchdog function.

```
        TA      EQU     0C7h

        ORG     00h
        SJMP    Start

        Org     30h
        Start :

        MOV     TA,     #0AAh   ;Timed
        MOV     TA,     #55h    ; Access
        ORL     IP,     #80h    ;Set RWT to restart the Watchdog Timer

        MOV     TA,     #0AAh   ;Timed
        MOV     TA,     #55h    ; Access
        ORL     PCON,   #44h    ;Set POR (PCON.6) bit for power on reset detect
                                ; and enable Watchdog Timer by setting EWT (PCON.2)
```

SECTION 11: INTERRUPTS

The Soft Micro family follows the standard 8051 convention for interrupts (with one extra) and is fully compatible. An interrupt stops the normal flow of processing and allows software to react to an event with special processing. This event can be external, time–related, or the result of serial communication. However, the interrupt will not be performed until the completion of the current instruction. This is discussed in more detail below. For each interrupt, there is an interrupt vector location. When an interrupt occurs, the CPU effectively performs a call to the corresponding vector address.

The interrupt vector is the location of the Interrupt Service Routine (ISR). Since the vector addresses are closely spaced, these ISRs typically use a jump to another more convenient location. An ISR performs special processing associated with the event that caused the interrupt. When the ISR is complete, the user returns control to the main program using an RETI instruction. This is the last instruction in an ISR and it performs two functions. First, it returns control to the main program. It returns to the instruction in the main program that would have been executed when the interrupt preempted it. Second, the RETI clears the pending interrupt condition. This allows the CPU to respond to other interrupts.

Each interrupt generally has an enable–control bit, a status flag bit, and a priority bit. Except for the new Power–fail Interrupt, the enable–control bits are located in the IE register and the priority bits are located in the IP register.The flags are scattered. Each interrupt aspect is discussed below.

There are six interrupt vector locations in a Soft Micro. Generally each interrupt has an associated vector location and flag. In the case of the Serial Interrupt, there are two sources with the same vector. In each case, a separate flag indicates the source of the event. The ISR knows the cause of the interrupt by the physical vector address that is called. For example, the External interrupt 0 vector is location 0003h, but the Timer 0 vector is 000Bh. Also note, the flags correspond to the event, not the interrupt. These flags will be activated even if a particular interrupt is not enabled so that software can poll the event. The flags (except serial port) are cleared when the CPU calls to the interrupt vector.

INTERRUPT SOURCE	VECTOR ADDRESS	FLAG	FLAG LOCATION
External Interrupt 0	0003h	IE0	TCON.1
Timer Interrupt 0	000Bh	TF0	TCON.5
External Interrupt 1	0013h	IE1	TCON.3
Timer Interrupt 1	001Bh	TF1	TCON.7
Serial I/O	0023h	RI & TI	SCON.0, SCON.1
Power Fail Warning	002Bh	PFW	PCON.5

INTERRUPT SOURCES

As shown above, there are two External Interrupts, two Timer Interrupts, two Serial Communication Interrupts, and a Power–fail Interrupt. To use an interrupt (except PFW), the software must globally enable the interrupt function. This is done with the EA bit (IE.7). Setting this bit to a logic 1 turns on the interrupt function. EA is cleared to a logic 0 by all resets. Next, each individual interrupt must be enabled. This is done using the other bits of the Interrupt Enable (IE) SFR. Each source has a corresponding bit that must be set to a logic 1. These are listed below.

INTERRUPT SOURCE	ENABLE BIT	LOCATION
External Interrupt 0	EX0	IE.0
Timer Interrupt 0	ET0	IE.1
External Interrupt 1	EX1	IE.2
Timer Interrupt 1	ET1	IE.3
Serial Port Interrupt	ES	IE.4
Power Fail Interrupt	EPFW	PCON.3

External Interrupts

The two External Interrupts are $\overline{INT0}$ and $\overline{INT1}$. They correspond to P3.2 and P3.3 respectively. These pins become interrupts when the respective interrupt is enabled as shown above. Otherwise, they are simply port pins. No other special action is required. Each pin is sampled once per machine cycle when the interrupts are enabled. $\overline{INT0}$ is enabled by setting the EX0 bit to a logic 1. $\overline{INT1}$ is enabled by setting the EX1 bit to a logic 1. These bits are located at IE.0 and IE.2 respectively. The External Interrupts each have a status flag that indicates that the condition has occurred. The flags are IE0 at TCON.1 and IE1 at TCON.3. These flags are set to a logic 1 when the interrupt condition occurs. They are cleared when the CPU calls to the appropriate interrupt vector.

The external interrupts can be programmed to respond to falling–edge or low–level activation. IT0 (TCON.0) and IT1 (TCON.2) control the edge/level nature of $\overline{INT0}$ and $\overline{INT1}$ respectively. When ITn is a logic 0, the associated interrupt is low–level activated. This causes the IEn flag to be set for as long as the $\overline{INTn}$ pin remains a logic 0. The interrupt (if enabled) will remain active during this period as well. Note that the level interrupt is not latched. Thus the pin must be held in a low state until the ISR can be activated. If the $\overline{INTn}$ pin is brought to a logic high prior to beginning the ISR, there will be no interrupt. If the $\overline{INTn}$ is left at a logic low after the RETI instruction of the ISR, another interrupt will be activated after one instruction is executed.

Setting the ITn bit to a logic 1 causes the external interrupt to be edge activated. This causes the micro to detect a falling edge on the $\overline{INTn}$ pin. This edge condition is latched until the interrupt is serviced. Thus in edge mode, the $\overline{INTn}$ pin can go from a logic 1 to a logic 0, then back to a logic 1 and the interrupt will still be active. After the falling–edge has been detected, the $\overline{INTn}$ pin is subsequently ignored until after the ISR is compete. The edge detector is actually a "pseudo–edge" detector. Since the pin is actually sampled, the condition must be a logic high for at least one machine cycle and logic low for at least one machine cycle in order to guarantee recognition of the falling edge. The IEn flag is automatically cleared when the interrupt is serviced.

Timer Interrupts

The Soft Micro, like the 8051, has two internal timers. These timers can each generate an interrupt when the value in the timer registers overflows. When the Timer 0 overflows, the TF0 flag is set to al logic 1. Likewise for the TF1 flag with respect to Timer 1. TF0 is located at TCON.5 and TF1 is located at TCON.7. These flags indicate the overflow condition. If the corresponding timer interrupt is desired, then ET0 at IE.1 and ET1 at IE.3 must be set to a logic 1 respectively. When set, the timer overflow will cause an interrupt to the appropriate vector location. If the interrupt is active, the flag will automatically be cleared by the CPU.

Serial Communication Interrupts

The on–chip serial port generates an interrupt when either a word is received or a word is transmitted. The interrupt is effectively a logical OR of the two conditions. Each condition has its own flag. The flags operate regardless of whether the interrupt has been enabled. RI is located at SCON.0 and represents a serial word received. TI is located at SCON.1 and represents a serial word transmitted. Each flag is set to a logic 1 to indicate an active state. Since there are two flags for one interrupt, these flags are used by the ISR to determine the cause of the interrupt. The flags must be cleared by software to clear the interrupt condition. The serial interrupt is activated by setting the ES bit at IE.4 to a logic 1.

Power–fail Warning Interrupt

The Soft Micro adds a new interrupt to the standard 8051 collection. It is used in conjunction with the power monitor and nonvolatile memory. During a power down or brown out, as V_{CC} is falling, the Soft Micro can generate an early warning Power–fail Interrupt (PFW). This allows the software to save critical data prior to entering a reset condition. Since the nonvolatile RAM is not affected by a reset, this data is effectively saved. Software can use the PFW to save the current routine, current data, shut off external functions, or simply to enter a know region of memory for the power down.

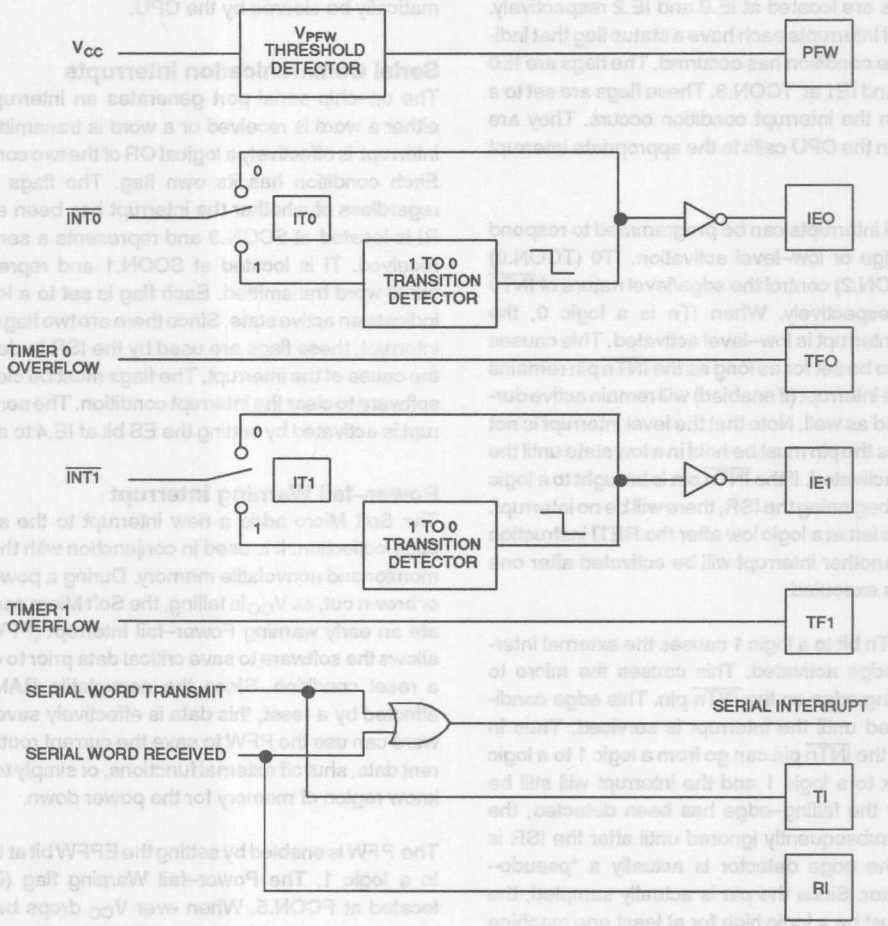
The PFW is enabled by setting the EPFW bit at PCON.3 to a logic 1. The Power–fail Warning flag (PFW) is located at PCON.5. When ever V_{CC} drops below the V_{PFW} voltage threshold, the PFW flag will be set to a logic 1. This flag will be cleared when read by software. If the voltage is still below the V_{PFW}, the flag will again be set immediately. This will occur regardless of whether the interrupt is enabled. The V_{PFW} voltage is different for each Soft Micro generation. Check the electrical specifications for details. Note that the PFW inter-

rupt is not controlled by the EA global enable bit. It can only be enabled or disabled using the EPFW bit.

Simulated Interrupts

Except for PFW, any interrupt can be forced by setting the corresponding flag to a logic 1 in software. This causes the micro to jump to the appropriate interrupt vector location. Clearing the appropriate flag manually will clear a pending interrupt. Note that the PFW flag can not be written by software.

INTERRUPT REQUEST SOURCES Figure 11–1



INTERRUPT ENABLE CONTROL BITS Figure 11–2

Bit Description:

All bits are read/write at any time and are cleared to 0 following any hardware reset.

IE.7: **EA**

"Enable All Interrupts": When set to 1, each interrupt except for PFW may be individually enabled or disabled by setting or clearing the associated IE.x bit. When cleared to 0, interrupts are globally disabled and no pending interrupt request will be acknowledged except for PFW.

IE.4: **ES**

"Enable Serial Interrupt": When set to 1, an interrupt request from either the serial port's TI or RI flags can be acknowledged. Serial I/O interrupts are disabled when cleared to 0.

IE.3: **ET1**

"Enable Timer 1 Interrupt": When set to 1, an interrupt request from Timer 1's TF1 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.2: **EX1**

"Enable External Interrupt 1": When set to 1, an interrupt from the IE1 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.1: **ET0**

"Enable Timer 0 Interrupt": When set to 1, an interrupt request from Timer 0's TF0 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

IE.0: **EX0**

"Enable External Interrupt 0": When set to 1, an interrupt request from the IE0 flag can be acknowledged. Interrupts are disabled from this source when cleared to 0.

sources to interrupt lower priority ISRs. The Power–fail Warning Interrupt automatically has the highest priority if enabled. The remaining interrupts can be programmed by the user to either high or low priority. The priority scheme woks as follows. The ISR for a low priority source can be interrupted by a high priority source. A low priority ISR can not be interrupted by another low priory source. Neither can a high priority ISR be interrupted by a another high priority source. The PFW source will interrupt any ISR if activated.

priority interrupt requests are received simultaneously, then the high priority source will be serviced. If two or more requests from equal priority sources are received, the following natural priority scheme will be used to arbitrate.

Each interrupt priority is determined by an individual bit as shown below. Setting the appropriate bit to a logic 1 will cause that interrupt to be high priority.

PRIORITY	FLAG	INTERRUPT SOURCE
1	PFW	Power–fail Warning
2	IE0	External Interrupt 0
3	TF0	Timer 0 Interrupt
4	IE1	External Interrupt 1
5	TF1	Timer 1 Interrupt
6	RI+TI	Serial I/O Interrupt

INTERRUPT PRIORITY CONTROL BITS Figure 11–3

Bit Description:

All bits are read/write at any time and are cleared to 0 following any hardware reset.

IP.4: **PS**

"Serial Port Priority": Programs Serial Port interrupts for high priority when set to 1. Low priority is selected when cleared to 0.

IP.3: **PT1**

"Timer 1 Priority": Programs Timer 1 interrupt for high priority when set to 1. Low priority is selected when cleared to 0.

IP.2: **PX1**

"Ext. Int. 1 Priority": Programs External Interrupt 1 for high priority when set to 1. Low priority is selected when cleared to 0.

IP.1: **PT0**

"Timer 0 Priority": Program Timer 0 interrupt for high priority when set to 1. Low priority is selected when cleared to 0.

IP.0: **PX0**

"Ext. Int. 0 Priority": Programs External Interrupt 0 for high priority when set to 1. Low priority is selected when cleared to 0.

INTERRUPT ACKNOWLEDGE

The various interrupt flags are sampled an latched once every machine cycle, specifically during clock phase S5P2 (see CPU timing section) regardless of other interrupt related activity. Likewise, the latched states of the flags are polled once every machine cycle for the sampling which took place during the previous machine cycle.

A complete interrupt acknowledge sequence consists of a total of four machine cycles, labeled as IA1, IA2, IA3, and IA4 in Figure 11–4. The various interrupt flags are sampled and latched once every machine cycle, specifically during clock phase S5P2. This is shown in the diagram as IA1. If one or more pending interrupt registers are latched, then during the following machine cycle (IA2) priority is resolved between one or more active interrupt requests.

Also during IA2, the hardware checks the state of the machine to insure that the following criteria are met before servicing the pending interrupt:

a) The current cycle is not part of an instruction within an interrupt service routine of an interrupt of equal or higher priority.

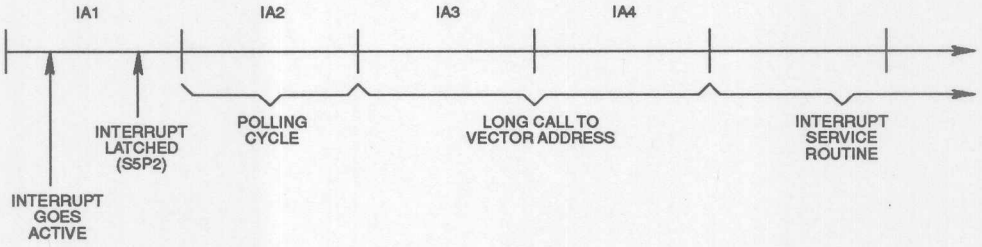
b) The current cycle is not the final machine cycle of an instruction which accesses the IP or IE registers.

If the above criteria are met during IA2, then a long call will be executed during IA3 and IA4 to the vector location of the pending interrupt source of highest priority and the interrupt acknowledge sequence will be complete. The vector locations for the various sources are summarized below.

FLAG	VECTOR ADDRESS	INTERRUPT SOURCE
PFW	002BH	Power Fail Warning
IE0	0003H	External Interrupt 0
TF0	000BH	Timer Interrupt 0
IE1	0013H	External Interrupt 1
TF1	001BH	Timer Interrupt 1
RI+TI	0023H	Serial I/O Interrupt

If the criteria during IA2 are not met, then the interrupt acknowledge sequence is aborted and the interrupt request latches will again be polled on the following machine cycle (which would have been IA3).

INTERRUPT ACKNOWLEDGE SEQUENCE Figure 11–4



The first criteria for the continuation of an interrupt acknowledge cycle is designed to maintain the priority relationship between interrupts and their priority level assignment. As a result, pending interrupt sources cannot be acknowledged during the execution of service routines of interrupts which are of equal or higher priority. Interrupt acknowledges are not allowed during an RETI instruction or during instructions which access IP or IE in order to insure that at least one more instruction will be executed before an interrupt is serviced.

The interrupt request flags are sampled and latched during every machine cycle regardless of the other interrupt activity on the device. Each time an attempt acknowledge takes place during IA2, it is based on the latched value of the flags during the previous machine

cycle. If the interrupt acknowledge does not take place for one of the reasons cited above, the request flag will become subsequently inactive and the interrupt will have been lost and will not be serviced.

When an interrupt request is acknowledged, a long call is executed to the interrupt vector location and the 2-byte return address is pushed onto the stack. In addition, an internal flag is set which indicates to the hardware the interrupt source that is being serviced. Execution then proceeds from the interrupt vector location. At the conclusion of the interrupt service routine, an RETI instruction should be performed to return control to the main program. The RETI performs the same action as a RET instruction in terms of its operation on the stack and the Program Counter. In other words, two bytes of return address are popped off the stack and loaded into the Program Counter. However, the RETI performs the additional operation of clearing the interrupt-in-service flag to inform the hardware that a service routine is no longer in progress. Therefore, an RETI should always be used to terminate an interrupt service routine. Failure to do so would indicate that the interrupt was still being serviced.

Higher priority interrupts, which are enabled, can interrupt lower priority interrupts. According to this rule, a higher priority interrupt could become pending just prior to machine cycle IA3 during an interrupt acknowledge of a lower priority interrupt. This would cause the hardware to vector to the higher priority service routine during the two machine cycles just after the long call to the lower priority interrupt so that no instruction within the lower priority interrupt service routine would have been executed.

SECTION 12: PARALLEL I/O

OVERVIEW

The Soft Microcontroller provides four 8–bit ports for general purpose I/O functions. They are called P0–P3. Each port pin is bit addressable, and each 8–bit port can be accessed as a whole byte. The ports are accessed using four Special Function Registers that control the respective port latch. All 32 port pins are bi–directional. Each bit has an associated latch (accessed via SFR), input buffer circuit, and output driver circuit. Ports 0, 2, and 3 also have alternate functions that can be used in place of general I/O.

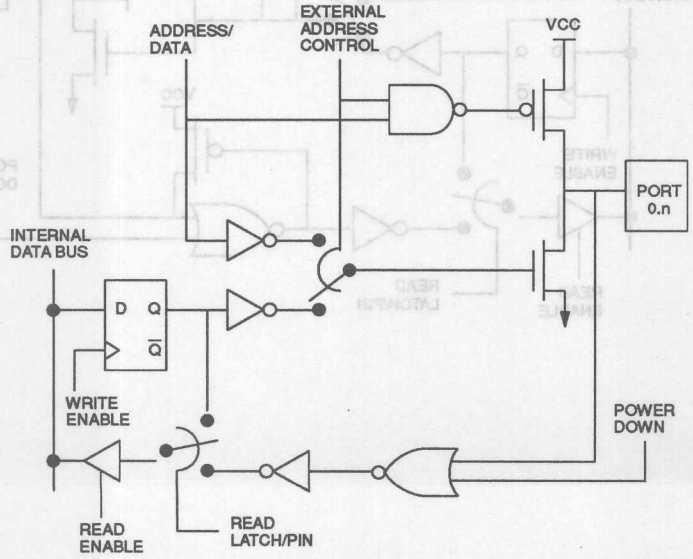
Ports 0 and 2 can serve as a multiplexed Expanded Memory bus for applications needing memory mapped I/O. This is discussed in the section on memory maps. In the DS5001 series devices, the Ports 0 and 2 can also serve as a slave interface to a host microprocessor. This function is discussed below. Port 3 pins each have individual, optional functions. The alternate Port 3 functions are shown below. It is unnecessary to take special action to convert these pins. Enabling the function will automatically convert the I/O pin into a special function. For example, enabling the serial port will automatically convert P3.0 and P3.1 into the RXD and TXD function. Also note that the special functions are independent. Enabling selected pins to perform their alternate job leaves the other as bit addressable I/O pins.

PIN	NAME	FUNCTION
P3.7	$\overline{RD}$	Expanded Data Memory Read Strobe
P3.6	$\overline{WR}$	Expanded Data Memory Write Strobe
P3.5	T1	Timer/Counter 1 Input
P3.4	T0	Timer/Counter 0 Input
P3.3	$\overline{INT1}$	External Interrupt 1 Input
P3.2	$\overline{INT0}$	External Interrupt 0 Input
P3.1	TXD	Serial Port Transmit Data
P3.0	RXD	Serial Port Receive Data

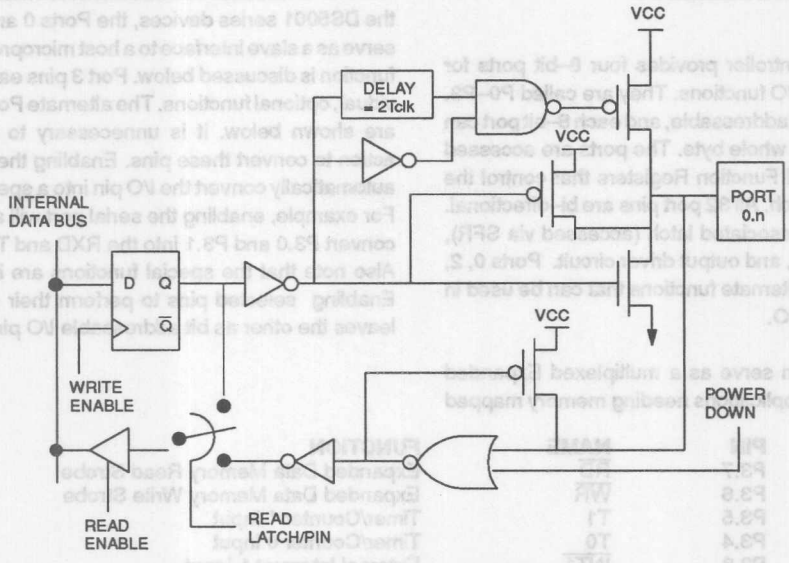
In order to use the alternate function for these Port 3 bits, the associated latch bit in the Port 3 Special Function must be set to a 1. Otherwise, the pin will be forced to a 0 condition. Port 1 has no alternate function for use in the system application; it is always available for parallel I/O functions.

All of the SFR latches for the parallel port pins are written with 1's during a hardware reset. Figure 12–1 illustrates functional circuit diagrams for bits within each of the four I/O ports.

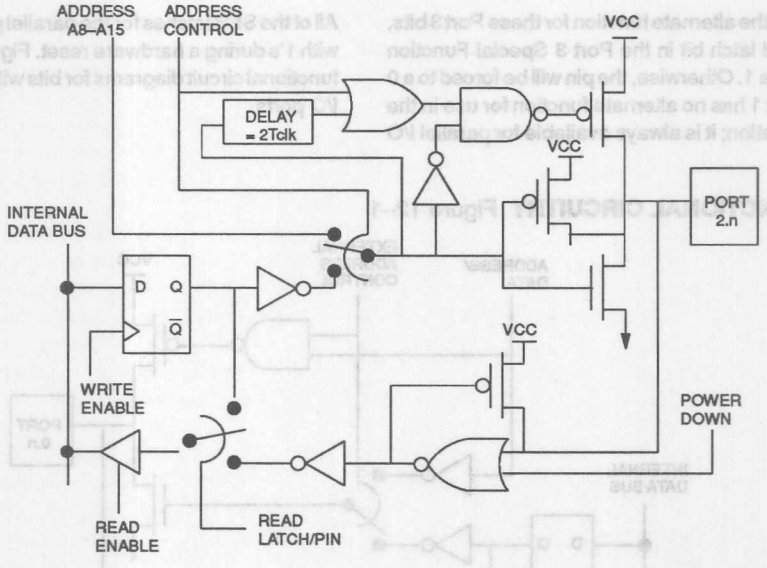
PORT 0 FUNCTIONAL CIRCUITRY Figure 12–1



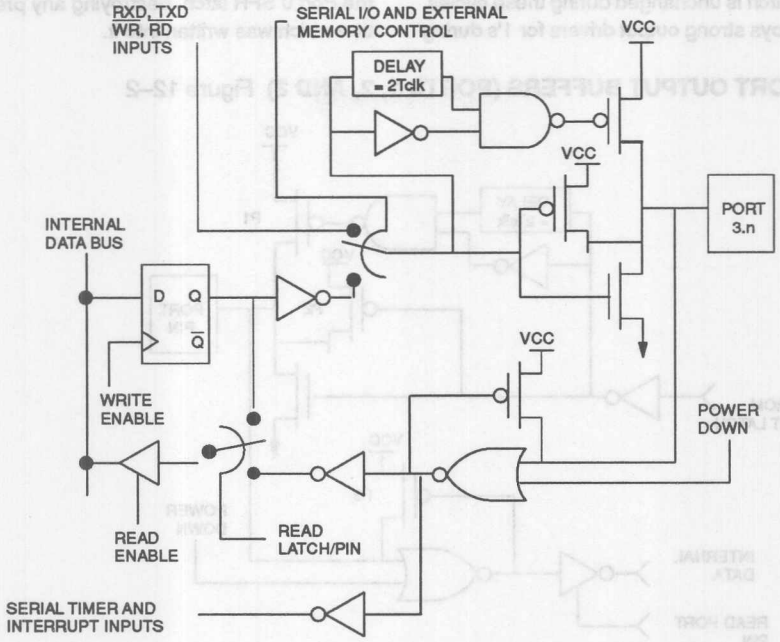
PORT 1 FUNCTIONAL CIRCUITRY



PORT 2 FUNCTIONAL CIRCUITRY



PORT 3 FUNCTIONAL CIRCUITRY



OUTPUT FUNCTIONS

Slightly different output buffer structures are implemented for the four parallel I/O ports. When the pins are used strictly for parallel I/O, ports 1, 2, and 3 have internal weak pull-up devices. Port 0, on the other hand, has a totem-pole output structure. When used as outputs, all port pins will drive the state to which the associated SFR latch bit has been set except for Port 0 which will only drive low. Port 0 requires a pull–up to drive high.

When an instruction is executed that writes a new value to the SFR latch for a parallel I/O port, the write actually occurs at S6P2 of the final machine cycle of the instruction. There is an additional delay in that the output buffers only sample the state of the latch's output during Phase 1 of any given clock period. As a result, the new value which is written to the latch will appear on the pin at S1P1 of the machine cycle following the final cycle of the instruction which performs the write to the port latch. See the section on CPU timing for clock details.

Port 1, 2, and 3 activate additional high–current pull-up devices when a write operation to the port necessitates a 0-to-1 transition on the I/O pin in order to speed up the transition time. The structure of these devices is illustrated in Figure 12–2. The pull-up structure is comprised of three pFET devices which are turned on when a logic 0 is applied to their gates and turned off when a 1 is applied. An n-channel device is used to drive a 0 on the pin and is turned on and off in the inverse sense of the pFET. When a 1 is applied, the n-channel FET is turned on and it is turned off when a 0 is applied.

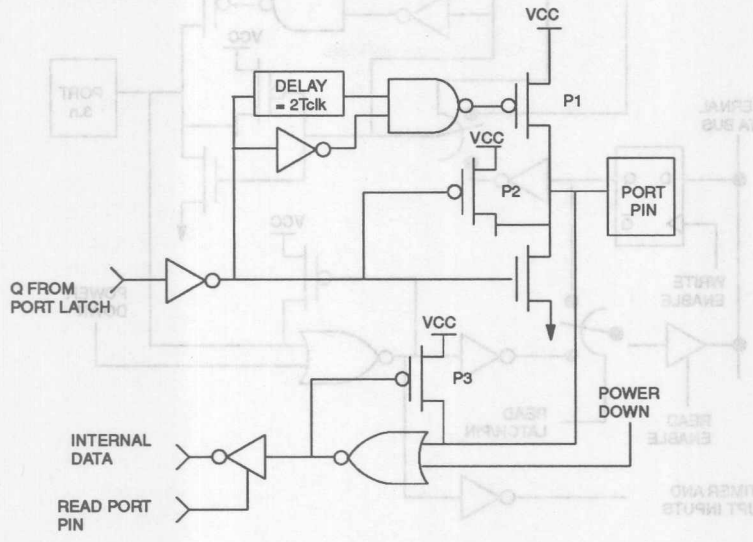
Following a 0-to-1 change in the state of the latch bit, transistor P1 will be turned on for two oscillator periods. This extra pull-up device can source about 10 mA (100 times more current than the normal P3 device). While P1 is turned on, it will in turn activate P3. The gate and P3 form a latch when P1 is turned off so that the state will be maintained on the pin.

P2 is a very weak pull-up device (about 1/10 the strength of P3) whose sole purpose is to restore a 1 to the pin should a negative glitch cause a 1 to be lost by forcing the latch to a 0 state.

When an access on the Expanded bus takes place, the pins of Port 0 and Port 2 are driven with address/data information. Port 2 outputs the most significant eight bits of address while Port 0 is time-multiplexed with the least significant eight bits of address and data. When 1's are output on Port 2 for address bits during these cycles,

strong current drivers are employed. The information in the Port 2 SFR latch is unchanged during these cycles. Port 0 also employs strong output drivers for 1's during these cycles. However, a value of 0FFH will be written to the Port 0 SFR latch, destroying any previous information which was written into it.

PARALLEL PORT OUTPUT BUFFERS (PORTS 1, 2, AND 3) Figure 12–2



INPUT FUNCTION

Any port pin can be used as a general purpose input by simply writing a logic 1 into the associated SFR latch. Ports 1, 2, and 3 have weak pull–ups, so they will go to a logic 1 state. However, the pull–up is sufficiently weak that an external circuit can easily overdrive it with a logic 0. Thus an output of 1 and an input are the same state. After setting the latch to a 1, the port can be read. If an external circuit drives high, reading the port will show a 1. If the external circuit drives low, the internal pull–up will be overcome and the pin will be low. Thus the read operation will see a logic 0. Port 0 is different in that it has no pull–up. Thus writing a 1 into the Port 0 latch causes the pin to tri–state. An external pull–up should be used. In the input state, the external circuit would overdrive the external pull–up on Port 0.

It can be seen in Figure 12–1 that there are actually two ways to read a port pin. The CPU can read the latch or the pin. These need not have identical values. A normal read instruction will read the state of the pin. It will neither read, nor modify the state of the latch. For example, if software writes the latch of Port 1 with an FFh, the port will output all high values, and also be configured as an input. If an external circuit pulls down the lower four bits, a read instruction would see F0h. The latch would still contain FFh. If the external circuit were to release the four lower bits, the port would return to the value of FFh.

There are a selected number of instructions that actually read the latch instead of the pin. These are called Read–Modify–Write instructions. These instructions read the state of the latch, possibly modify it, then write the result back to the latch. The Read–Modify–Write instructions are listed below.

READ-MODIFY-WRITE INSTRUCTIONS

MNEMONIC		DESCRIPTION
ANL	–	Logical AND
ORL	–	Logical OR
XRL	–	Logical Exclusive OR

JBC	–	Branch if Bit Set and Clear (bit)
CPL	–	Complement Bit
INC	–	Increment
DEC	–	Decrement
DJNZ	–	Decrement and Branch if not Zero
MOV PX.n,C	–	Move Carry Bit to bit n of Port X
CLR PX.n	–	Clear bit n in Port X
SETB PX.n	–	Set bit n in Port X

Read-Modify-Write instructions input the state of the latch rather than the pin so that the operation takes place on the value which was originally written to the latch by the software.

REPROGRAMMABLE PERIPHERAL CONTROLLER (RPC)

The Reprogrammable Peripheral Controller (RPC) mode of the DS5001 series emulates the 8042 slave hardware interface commonly used in IBM–compatible PCs for control of peripherals such as a keyboard or a mouse device. In addition to a direct interface to the PC backplane bus, the DS5001 brings the advantages of up to 128K of reprogrammable, nonvolatile program and data memory to intelligent peripheral control. The nonvolatile data memory accessed by the DS5001 can be used for system configuration, hard disk setup parameters, or even maintenance records. System peripheral developers now have the benefit of programming in the standard 8051 instruction set with its more powerful features and wider development support.

In operating as a slave controller, the DS5001 provides communication with a host processor via three resource registers: Data Bus Buffer In (DBBIN), Data Bus Buffer Out (DBBOUT), and Status (STATUS). The host may read data or status and write data or commands. The STATUS register provides information about DBBIN, DBBOUT, and user–defined flags. Both DBBIN and DBBOUT share special function register address 80H with Port 0. The context will determine which register is used. The STATUS register is at SFR location 0DAH.

To enable the RPC mode, the RPCON bit in the RPCTL register (describe below) must be set to a 1. At this time, Ports 0 and 2 are reconfigured to emulate the 8042 hardware interface as shown in Figure 12–3. Port 0 becomes an 8–bit data bus that can connect directly to a PC data bus. Port 2 provides the control and address information for the data bus. Both ports are true bidirectional I/O devices in this mode. Normal operation of these ports is suspended when RPC mode is enabled. The modified port functions are described as follows:

Port 0: D0–7 This is the 8–bit data bus of the RPC. As a bidirectional I/O bus, it can interface directly to a PC or other host.

Port 2.0: A0 Address input used to determine whether the data bus word is data or command/status.

Port 2.1: $\overline{CE}$ If a multiple RPC mode environment is required, this input can be used to select an individual DS5001 on a common bus.

Port 2.2: $\overline{RD}$ Input that allows the host to read data or status from the DBBOUT or STATUS.

Port 2.3: $\overline{WR}$ Input that allows the host to write data or commands to DBBIN.

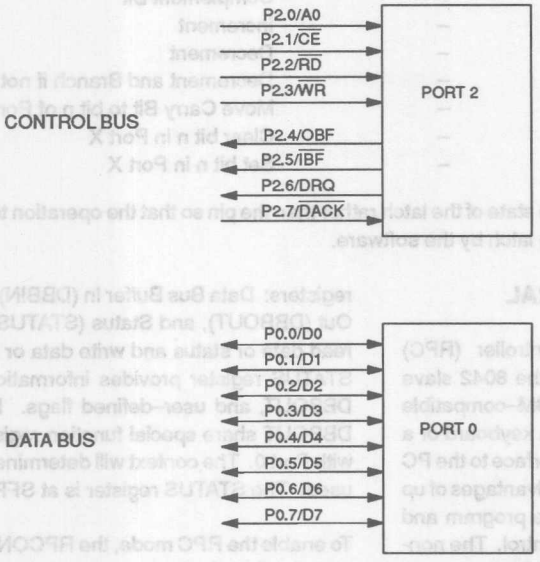
Port 2.4: OBF Output flag that indicates to a host that the output buffer is full and should be read.

Port 2.5: $\overline{IBF}$ Output that indicates to a host that the input buffer is empty.

Port 2.6: DRQ Output that indicates to a host that a DMA is required.

Port 2.7: $\overline{DACK}$ Input that indicates to the DS5001 that the host has granted a DMA.

USE OF THE RPC MODE Figure 12–3



USE OF THE RPC MODE Figure 12–4

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A0	REGISTER
0	0	1	0	DATA OUT
0	0	1	1	STATUS
0	1	0	0	DATA IN
0	1	0	1	COMMAND IN
1	X	X	X	NO REGISTER

RPC INTERRUPTS

RPC mode provides an additional interrupt to the standard DS5000 set. An Input Buffer Full Interrupt (IBF) will be performed (if enabled) when data is written to the DBBIN from a host. When enabled, this interrupt replaces the Timer 1 interrupt (vector location 1BH). Regardless of whether this interrupt is enabled, future writes are locked out until the DBBIN is ready by the DS5001. The DS5001 provides two outputs to interrupt the host system as needed. These are Output Buffer Full (OBF) and Input Buffer Empty ($\overline{IBF}$).

RPC STATUS REGISTER – STATUS (ADDRESS 0DAH) Figure 12–5

ST7	ST6	ST5	ST4	IAO	FO	IBF	OBF

Bit Description:

RPS.7–4: General purpose status bits that can be written by the DS5001 and can be read by the external host.

Initialization: Cleared when RPCON=0.

Read Access: Can be read by the DS5001 and host CPU when RPC mode is invoked.

Write Access: Can be written by the DS5001 when RPC mode is invoked.

RPS.3: **IA0**
Stores the value of the external system A0 for the last DBBIN Write when a valid write occurs (as determined by the IBF flag).

Initialization: Cleared when RPC=0.

Read Access: Can be read by the DS5001 and host CPU when in RPC mode.

Write Access: Automatically written when a valid DBBIN Write occurs. Cannot be written otherwise.

RPS.2: **F0**
General purpose flag written by the DS5001 and read by the external host.

Initialization: Cleared when RPC=0.

Read Access: Can be read by the DS5001 and host CPU when in RPC mode.

Write Access: Can be written by the DS5001 when in RPC mode.

RPS.1: **IBF**
Input Buffer Full Flag is set following a write by the external host, and is cleared following a read of the DBBIN by the DS5001.

Initialization: Cleared when RPC=0.

Read Access: Can be read by the DS5001 and host CPU when in RPC mode.

Write Access: Written automatically as part of the RPC communication. Cannot be set by the application software.

RPS.0: **OBF**
Output Buffer Full Flag is set following a write of the DBBOUT by the DS5001, and is cleared following a read of the DBBOUT by the external host.

Initialization: Cleared when RPC=0.

Read Access: Can be read by the DS5001 and host CPU when in RPC mode.

Write Access: Written automatically as part of the RPC communication. Cannot be set by the application software.

RPC PROTOCOL

Data is written to the DS5001 by the host CPU and is placed in the DBBIN. At this time, the IBF flag is set in the RPC Status Register. If enabled by the IBI bit in the RPCTL register, an IBI interrupt will occur. No further updates of the DBBIN will be allowed until the buffer is read by the DS5001. Once read, the IBF flag will be cleared. When the DBBOUT is written to by the DS5001, the OBF is set in the RPC Status Register (STATUS). No future writes are allowed until the DBBOUT is read by the external host. The OBF is cleared when such a read takes place.

The RPC mode provides a simple interface to a host processor. In general, four control bits specify the operation to be performed. This works as shown in Figure 12–3.

These conditions provide the basis of a complete slave interface. The protocol for such communications might operate as follows:

1. Host processor reads STATUS.
2. If DBBIN is empty (IBF=0), host writes a data or command word to DBBIN.
3. If DBBOUT is full (OBF=1), host reads a word from DBBOUT.
4. RPC detects IBF flag via interrupt or polling. Input data or command word is processed.
5. RPC recognizes OBF=0, and writes a new word to DBBOUT.

Timing diagrams in RPC AC electrical specifications illustrate the operation of the RPC mode bus transfers. A DBBOUT read places the contents of DBBOUT on the data bus and clears OBF. A STATUS register read places the contents of the STATUS register on the data bus. A write to DBBIN causes the contents of the data bus to be transferred to the DBBIN, and the IBF flag (STATUS) is set. A command write operates in the same way. The DS5001 can determine whether the write was data or command by examining the IA0 bit in the STATUS register. This bit will be equal to the A0 input of the most recent valid host write operation.

DMA OPERATION

If DMA transfers are required, the DS5001 RPC mode can support them. DMA transfers are initiated by setting the DMA bit in the RPCTL register. The DRQ output is de–asserted at this time. DRQ can be asserted by writing a 1 to the DRQ line (P2.6) from software. The host CPU must respond by pulling the $\overline{\text{DACK}}$ input low. Data can then be transferred according to the user's required protocol. DMA mode can be cancelled by clearing the DMA bit, by a DS5001 reset, or by clearing the RPCON bit of the RPCTL control register to leave the RPC mode.

RPC CONTROL REGISTER – RPCTL (ADDRESS 0D8H) Figure 12–6

RNR	–	EXBS	AE	IBI	DMA	RPCON	RGO

Bit Description:

RPCTL.3: **IBI**
When using the RPC mode, an interrupt may be required for the Input Buffer Flag. This interrupt is enabled by setting the Input Buffer Interrupt (IBI) bit. At this time, the timer 1 interrupt is disabled, and this RPC mode interrupt is used in its place (vector location 1BH). This bit can be set only when the RPCON bit is set.

Initialization: Cleared on all resets, and when the RPCON bit is cleared.

Read Access: Can be read at any time.

Write Access: Can be written when RPC mode is enabled (RPCON=1).

RPCTL.2: **DMA**
This bit is set to enable DMA transfers when RPC mode is invoked. It can only be set when RPCON=1.

Initialization: Cleared on all resets, and when the RPC is cleared.

Read Access: Can be read at any time.

Write Access: Can be written when RPC mode is enabled (RPCON=1).

RPCTL.1: **RPCON**
Enable the 8042 I/O protocol. When set, Port 0 becomes the data bus, and Port 2 becomes the control signals as shown in Figure 12–3.

Initialization: Cleared on all resets.

Read Access: Can be read at any time.

Write Access: Can be written at any time.

SECTION 13: PROGRAMMABLE TIMERS

FUNCTIONAL DESCRIPTION

The Soft Micro incorporates two 16-bit timers called Timer 0 and Timer 1. Both can be used to generate precise time intervals, measure external pulse widths, or count the number of externally applied pulses.

Each programmable timer operates either as a "timer" in which time interval interrupts may be generated or as a "counter", in which the timer register is incremented when transitions are detected on an external input pin.

When a programmable timer is operating as a "timer", the least-significant timer register is incremented once every machine cycle or at 1/12 the frequency of the clock oscillator. When a 12 MHz crystal is used, the register will be incremented once every 1 µs.

When "counter" operation is selected, the least-significant timer register is incremented each time that a 1-to-0 transition is detected on the corresponding input pin that may be assigned for the timer (T0 for Timer 0, T1 for Timer 1). These pins are the optional function of P3.4 and P3.5 respectively. The timing of the "counter" mode is internally synchronized to the machine cycles. During S5P2 of every machine cycle, the external input pin is sampled. A 1-to-0 transition is defined as a 1 detected during a machine cycle followed by a 0 detected in the S5P2 clock phase of the next machine cycle. The new count value in the timer register will be present during clock phase S3P1 of the next successive (or third) machine cycle. See the section on timing for details.

The TMOD and TCON Special Function registers are used to control the initialization of the two programmable timers. A summary of the bits contained in TMOD is shown in Figure 13–1. The relevant TCON register bits are depicted in Figure 13–2. Each Timer has four control bits associated with it including C/$\overline{T}$, GATE, M1, and M0. C/$\overline{T}$=1 selects counter operation and C/$\overline{T}$=0 selects timer operation.

A separate GATE bit in the TMOD register is provided for each timer. These bits enable an associated external interrupt input pin as a gating control for the timer or counter function. The P3.2 ($\overline{INT0}$) pin operates in conjunction with Timer 0 while the P3.3 ($\overline{INT1}$) pin operates with Timer 1. When the Timer Run bit (TRn) and GATE are both set to a 1, the timer or counter function will be enabled only during the times that the associated interrupt input pin is at a 1 level. When the Timer function is selected, the GATE bit provides a means of measuring the widths of logic 1 pulses applied to the interrupt pin in units of machine cycles. When the counter function is selected, the pulse is measured in units of 1-to-0 transitions detected on the external counter input pin.

Both of the programmable timers have M1,M0 control bits in the TMOD register which are used to select one of the four operating modes described below.

TMOD REGISTER CONTROL BIT SUMMARY Figure 13–1

Bit Description

TMOD.7 (Timer 1); TMOD.3 (Timer 0): **GATE**

"Gate Control": When set to 1 with TRns=1, timer/counter's input count pulses will only be delivered while a 1 is present on the $\overline{INT}$ pin. When cleared to 0, counter pulses will always be received by the timer/counter as long as TRn=1.

Initialization: Cleared to 0 on any reset.

TMOD.6 (Timer 1); TMOD.2 (Timer 0): **C/$\overline{T}$**

"Counter/Timer Select": When set to a 1, the counter function is selected for the associated programmable timer; when cleared to 0, the timer function is selected.

Initialization: Cleared to 0 on any reset.

TMOD.5, TMOD.4: **Timer 1 Mode Control**

"Mode Select" These bit select the operating mode of the associated timer/counter as follows:

M1	M0	
0	0	Mode 0: Eight bits with 5-bit prescale
0	1	Mode 1: 16 bits with no prescale
1	0	Mode 2: Eight bits with auto-reload
1	1	Mode 3: Timer 0 – Two 8-bit timers Timer 1 – Stopped

Initialization: Cleared to 0 on any reset.

TMOD.1, TMOD.0: **Timer 0 Mode Control**

"Mode Select" These bit select the operating mode of the associated timer/counter as follows:

M1	M0	
0	0	Mode 0: Eight bits with 5-bit prescale
0	1	Mode 1: 16 bits with no prescale
1	0	Mode 2: Eight bits with auto-reload
1	1	Mode 3: Timer 0 – Two 8-bit timers Timer 1 – Stopped

Initialization: Cleared to 0 on any reset.

TCON REGISTER CONTROL/STATUS BITS Figure 13–2

Bit Description:

TCON.7: **TF1**

"Timer 1 Overflow Flag": Status bit set to 1 when Timer 1 overflows from a previous count value of all 1's. Cleared to 0 when CPU vectors to Timer 1 Interrupt service routine.

Initialization: Cleared to 0 on any type of reset.

TCON.6: **TR1**

"Timer 1 Run Control": When set to a 1 by software, Timer 1 operation will be enabled. Timer 1 is disabled when cleared to 0.

Initialization: Cleared to 0 on any type of reset.

TCON.5: **TF0**

"Timer 0 Overflow": Status bit set to 1 when Timer 0 overflows from a previous count value of all 1's. Cleared to 0 when CPU vectors to Timer 0 interrupt service routine.

Initialization: Cleared to 0 on ay type of reset.

TCON.4: **TR0**

"Timer 0 Run Control": When set to a 1 by a software, Timer 0 operation is enabled. Timer 0 is disabled when cleared to 0.

Initialization: Cleared to 0 on any type of reset.

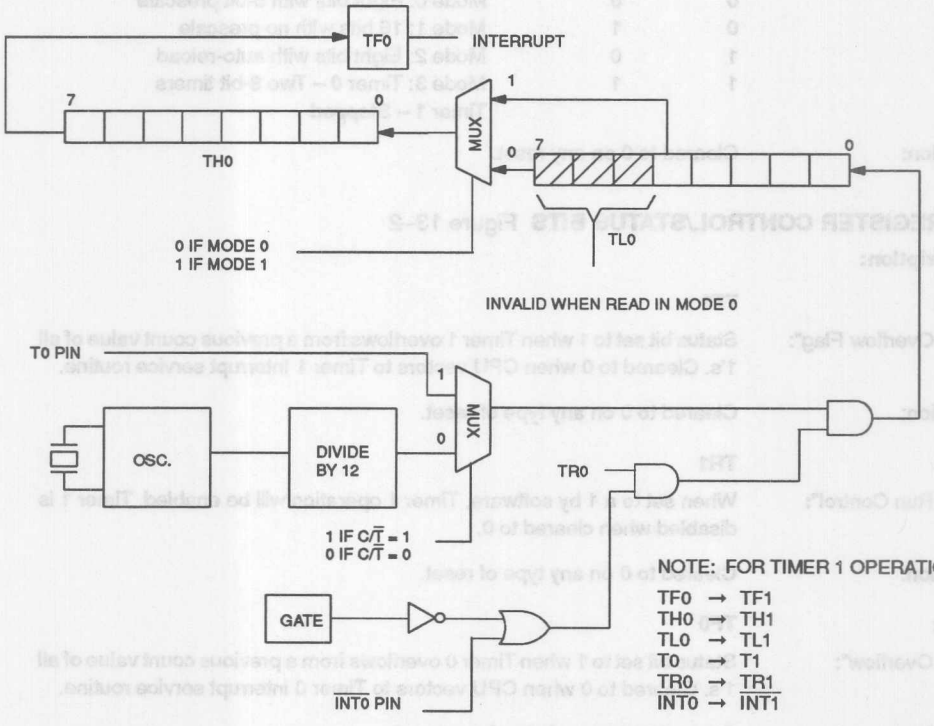
Mode 0

Figure 13–3 is a block diagram of a timer/counter operating in Mode 0. Mode 0 configures either programmable timer for operation as a 13-bit timer/counter. For Timer 0, selection of Mode 0 configures bit 4 – 0 of TL0 as bits 4 – 0 respectively of the 13-bit timer/counter register. In addition, bits 7 – 0 of TH0 are configured as bits 12 – 5 respectively of the 13-bit timer/counter register. Bits 7 – 5 of TL0 are indeterminate and should be ignored when read. All of the timer/counter bits are cleared to 0 by a hardware reset. When the TR0 bit is set with either GATE=0 or $\overline{INT0}$=1, the 13-bit register will be incremented as each count is received. The previous value of the 13-bit register is unchanged when the TR0 bit is set to a 1 from a previous 0 condition.

When the 13-bit timer/counter reaches a value of 1FFFH (all 1's) the next count received will cause the value to roll over to 0000 and the TF0 flag will be set. Additionally, an interrupt will be generated if it had been enabled.

Mode 0 operation for Timer 1 is functionally identical to that described for Timer 0. TH1 and TL1 are used to form the 13-bit register as just described for Timer 0. Likewise, TR1, TF1, and INT1 perform the functions described for TR0, TF0, and $\overline{INT0}$.

TIMER/COUNTER MODE 0 AND 1 OPERATION Figure 13–3



Mode 1

Mode 1 for both programmable timers operates in an identical fashion described for Mode 0, except Mode 1 configures a 16-bit timer/counter register. In this case, for Timer 0, TH0 contains the most significant eight bits of the count value while TL0 holds the least significant eight bits. Timer 1 uses TH1, TL1 in an identical fashion in Mode 1. Figure 13–3 is also a diagram depicting operation in Mode 1 for the timer/counters.

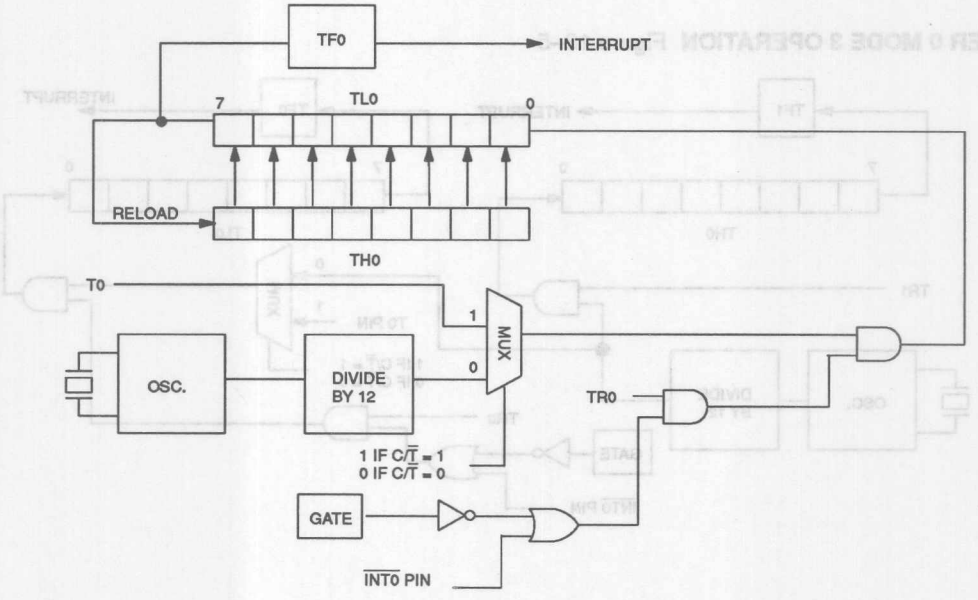
Mode 2

The selection of Mode 2 configures an 8-bit timer/counter with automatic reload of a value preset by software.

Figure 13–4 illustrates a functional block diagram of this operational mode. When Timer 0 is used in Mode 2, TL0 is incremented as each count is received. When the value of 0FFH (all 1's) is reached, TF0 will be set on the next count and the reload value held in TH0 will be transferred into TL0. TH0 remains unchanged until it is modified by the application software.

Timer 1 operates in an identical fashion when it is set for operation in Mode 2.

TIMER/COUNTER MODE 2 OPERATION Figure 13–4



Mode 3

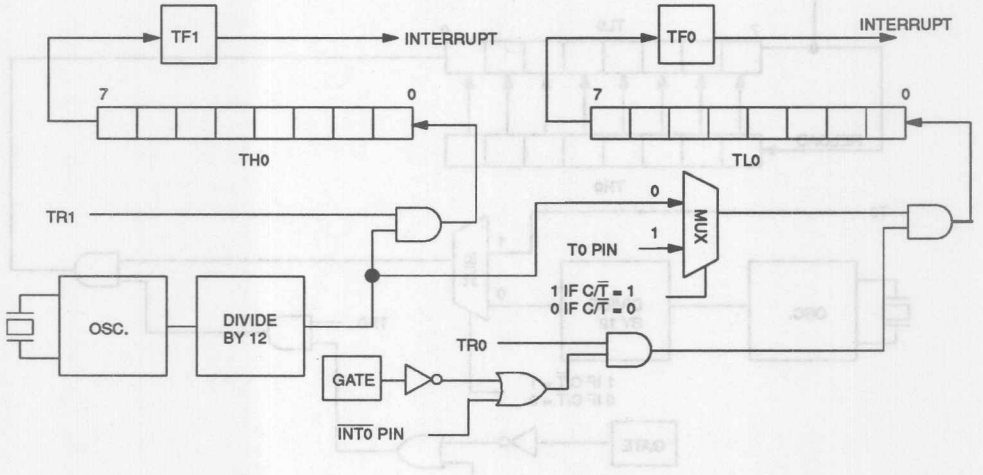
When Timer 0 is selected for operation in Mode 3, both TH0 and TL0 are configured independently as an 8-bit timer/counter and as an 8-bit timer. Figure 13–5 illustrates the function of Timer 0 for Mode 3 operation.

For Timer 0 in Mode 3, TL0 becomes an 8-bit timer/counter which is controlled by the Timer 0 control bits (TR0 and TF0) in the TMOD and TCON registers. TL0's count input may be assigned to either the 12 t_{CLK} signal or to the external T0 pin via C/$\overline{T}$ for Timer 0. In addition, the count input may be gated as a function of the $\overline{INT0}$ pin using Timer 0's GATE bit in TMOD.

TH0 becomes an 8-bit timer when Mode 3 is selected for Timer 0. TH0's input can only be the 12 t_{CLK} signal. TR1 and TF1 are assigned for use with TH0 as is the interrupt for Timer 1, which will be generated when TH0 overflows from all 1's.

When Timer 1 is selected for operation in Mode 3, it stops counting and holds its current value. This action is the same as setting TR1=0. When Timer 0 is selected in Mode 3, Timer 1's control bits are stolen as described above. As a result, Timer 1's functions are limited in this MODE. It is forced to operate as a timer whose clock input is 12 t_{CLK} and it cannot generate an interrupt on overflow. In addition, it also cannot be used with the GATE function. However, it can be started and stopped by switching it into or out of Mode 3 or it can be assigned as a baud rate generator for the serial port.

TIMER 0 MODE 3 OPERATION Figure 13–5



SECTION 14: SERIAL I/O

FUNCTION DESCRIPTION

The Soft Microcontroller, like the 8051, includes a powerful Serial I/O (UART) port capable of both synchronous and asynchronous communication. It is fully programmable for baud rate and time–base source. The serial port is constructed using P3.0 as Receive Data (RXD) and P3.1 Transmit Data (TXD). Note that no special action other than enabling the function is required to make these pins become the serial port. The Serial I/O port is capable of full duplex operation in asynchronous mode and half–duplex operation in synchronous mode.

The Serial Port consists of a receive shift register, receive buffer, transmit shift register and control logic. An incoming serial word from an external source is shifted into the receive shift register one bit at a time. Bits are shifted at the baud rate, which is programmable. The baud rate must be programmed by user's software to match the incoming frequency or the serial data will be unintelligible. Once the word is received, the UART transfers it into the receive buffer. At this time, the UART can receive another byte into its shift register. Once a word is received, the software should read the receive buffer before another word is completely received. The UART will automatically transfer the new word into the receive buffer regardless of whether software has read the old value. This destroys the data that had been present from the previous word. At 9600 baud, receiving an asynchronous word takes 1.04 ms. Thus software must read a received word within 1.04 ms or it may be overwritten by another incoming word.

The transmit shift register has no buffer. Software writes into the shift register and the word is immediately shifted out. Thus software must wait until the serial word is shifted out before writing another to transmit. Both the receive buffer and the transmit shift register are located in the SFR map. Furthermore, they reside at the same address called SBUF (99h). Reading SBUF automatically transfers the word out of the serial receive buffer. Writing to SBUF automatically transfers a byte into the transmit shift register. Serial Port operation is controlled via the SCON (98h) register.

Each UART function (receive and transmit) is capable of generating an interrupt. If enabled, the receive function interrupts the CPU when a word has been shifted in. This indicates that software should read the receive buffer. The UART will set the RI flag bit in the SCON.0 location to indicate the source of the interrupt. The UART will also generate an interrupt when it has completely shifted out a word. This indicates that another word can be transmitted. The UART will set the TI flag bit at SCON.1 to indicate the source of the interrupt. Remember that the Serial Interrupt vectors to location 23h regardless of the source. The ISR must determine the cause of the interrupt from the flags mentioned above.

SERIAL I/O OPERATING MODES Table 14–1

MODE	SYNC/ASYNC	BAUD CLOCK	DATA BITS	START/STOP	9TH DATA BIT FUNCTION
MODE 0	SYNC	12 t_{CLK}	8	None	None
MODE 1	ASYNC	Timer 1 Overflow	8	1 Start 1 Stop	None
MODE 2	ASYNC	32 t_{CLK} or 64 t_{CLK}	9	1 Start 1 Stop	0, 1, or parity
MODE 3	ASYNC	Timer 1 Overflow	9	1 Start 1 Stop	0, 1, or parity

The Soft Microcontroller UART has four modes (Mode 0–3) of operation as shown in Table 14–1. Mode 0, is a synchronous mode. This means that the microcontroller UART will supply a clock to synchronize the data I/O shifting. One clock pulse is generated per bit. The external device that is communicating with the micro must also use this clock. This mode is typically used with serial peripherals. Synchronous mode is generally capable of a higher speed communication speeds than the asynchronous modes. It generates its speed as a fixed function of 12 microcontroller oscillator clocks per bit.

Mode 1 is a 10–bit asynchronous mode using 8–bit words, one start bit, and one stop bit. The time base is generated from the Timer 1 overflow and is therefore fully programmable. A user simply loads the timer with a value that generates the required time interval at its

overflow. This is the most common mode of communicating with a PC COM port or similar device. When talking to a PC in Mode 1, the PC would be set to 8N1 or 8 bits, no parity, 1 stop. Common baud rates are 2400, 9600, and 19200 bps. The Soft Micro can communicate as fast as 57,600 bps in Mode 1.

Mode 2 is an 11–bit asynchronous mode using 8 or 9–bit words and one stop bit. The time base offers a choice of two fixed relationships of either 32 or 64 oscillator clocks per bit. It is not otherwise programmable in speed. The 9th bit is selected manually. It can be set to a 1, 0, or parity. Thus Mode 2 could appear to have two stop bits by selecting the 9th bit to be a logic 1.

Mode 3 is similar to Modes 1 and 2. Like Mode 2, it uses 9–bit words instead of 8. Also like Mode 2, the 9th bit can be either 0, 1, or parity. Like Mode 1, it uses the Timer 1 mechanism to generate baud rates. This mode can be used with a PC COM port set for 8N2 (8 bits, no parity, two stop bits) by setting the 9th bit to a 1. It can also support 8E1 (8bits, even parity, one stop). Parity is done by transferring the parity bit (PSW.0) to the 9th bit of the UART (SCON.3). Since the Soft Micro CPU sets the parity bit to indicate an odd number of bits in the accumulator, a 9–bit serial word containing this parity bit would have even parity.

The UART is controlled by the SCON register at SFR location 98h. These bits are described below in Figure 14–1. The Soft Micro UART begins transmission after software writes to the SBUF register. Data is always shifted out with the LSB first. Each mode is discussed in detail below following Figure 14–1.

SERIAL PORT CONTROL REGISTER Figure 14–1

Bit Description:

SCON.7, SCON.6: **SM0, SM1**

"Mode Select": Used to select the operational mode of the serial I/O port as follows:

SM0	SM1	MODE	FUNCTION	WORD LENGTH	BAUD CLOCK PERIOD
0	0	Mode 0	Sync	8 bits	12 t_{CLK}
0	1	Mode 1	Async	10 bits	Timer 1 Overflow
1	0	Mode 2	Async	11 bits	64 t_{CLK} or 32 t_{CLK}
1	1	Mode 3	Async	11 bits	Timer 1 Overflow

Initialization: Cleared to a 0 on any type of reset.

SCON.5: **SM2**

"Multiple MCU Comm.": Used to enable the multiple microcontroller communications feature for Modes 2 and 3. When SM2=1, RI will be activated only when serial words are received which cause RB8 to be set to 1.

Initialization: Cleared to a 0 on any type of reset.

SCON.4: **REN**

"Receive Enable": When set to 1, the receive shift register will be enabled. Disabled when cleared to 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.3: **TB8**

"Xmit Bit 8": Can be set or cleared to define the state of the 9th data bit in Modes 2 and 3 of a serial data word.

Initialization: Cleared to a 0 on any type of reset.

SCON.2: **RB8**

"Rcv. Bit 8": Indicates the state of the 9th data bit received while in Mode 2 or 3 operation. If Mode 1 is selected with SM2=0, RB8 is the state of the stop bit which was received. RB8 is not used in Mode 0.

Initialization: Cleared to a 0 on any type of reset.

SCON.1: **TI**

"Xmit Interrupt": Status bit used to signal that a word has been completely shifted out in Mode 0; it is set at the end of the 8th data bit. Set when the stop bit is transmitted.

Initialization: Cleared to a 0 on any type of reset.

SCON.0: **RI**

"Receive Interrupt": Status bit used to signal that a serial data word has been received and loaded into the receive buffer register. In Mode 0, it is set at the end of the 8th bit time. It is set at the mid-bit time of the incoming stop bit in all other modes of a valid received word according to the state of SM2.

Initialization: Cleared to a 0 on any type of reset.

BAUD RATE GENERATION

As shown in Table 14–1, the baud rate clock source for the serial I/O is determined by the selection of the operating mode.

In Modes 0 and 2, the baud rate is divided down from the clock oscillator frequency by a fixed value. In Mode 0, the baud rate is 1/12 of the clock oscillator frequency, or:

$$\text{Mode 0 Baud Rate} = \frac{1}{12t_{CLK}}$$

In Mode 2, the baud rate is either 1/32 or 1/64 of the clock oscillator frequency. This selection is dependent on the state of the SMOD bit (PCON.7). If SMOD=0, the baud rate will be 1/64 the clock oscillator frequency. If SMOD=1, the baud rate is 1/32 the clock oscillator frequency. This can also be given as:

$$\text{Mode 2 Baud Rate} = \frac{2^{SMOD}}{64} \times \frac{1}{t_{CLK}}$$

Note that 2^{SMOD} means two to the power of SMOD. $2^0=1$, $2^1=2$

The baud rates in Modes 1 and 3 are variable as they are generated as a function of the Timer 1 overflow signal and the value of the SMOD bit. A general equation which describes the baud rate frequency can be given as:

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{1}{t_{T1}}$$

where, t_{T1} is the overflow period of Timer 1. In this application Timer 1 can be configured in either the timer or the counter configuration. If the counter configuration is selected, then the baud rate frequency will be divided down from an external clock source applied to the P3.3 ($\overline{INT1}$) pin. As a general guideline, the GATE bit for Timer 1 should be 0 if the counter function is selected in this situation so that a continuous clock source will be available for the baud generator.

In most applications, Timer 1 will be configured as a timer which uses the internal clock oscillator frequency as its clock source. The baud rate will then be divided down from the time base applied to the XTAL1 and XTAL2 pins. In order to provide the most flexibility, Timer 1 should be programmed to operate in Mode 2 which configures TL1 as an 8-bit timer which is automatically reloaded with the value held in TH1 when its timeout condition is reached. This operational mode is selected by assigning the TMOD register control bits in the following configuration:

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
0	0	1	0	X	X	X	X

In the configuration selected above, the baud rate for the serial port can be expressed as:

$$\text{Serial I/O Mode 1, 3 Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{1}{12t_{CLK}\,(256 - (TH1))}$$

Table 14–2 lists some commonly used baud rates which can be derived by using Timer 1 in the timer configuration described above with a 11.059 MHz crystal as the time base.

TIMER 1 BAUD RATE GENERATION Table 14–2

BAUD RATE (BPS)	$1/t_{CLK}$ (MHz)	SMOD (SCON.7)	TIMER 1 C/$\overline{T}$	TIMER MODE	TH1
19200	11.059	1	0	2	0FDH
9600	11.059	0	0	2	0FDH
4800	11.059	0	0	2	0FAH
2400	11.059	0	0	2	0F4H
1200	11.059	0	0	2	0E8H
300	11.059	0	0	2	0A0H

When Timer 1 is used in this manner its interrupt should be disabled since the timeout period is much faster than is reasonable for interrupt response and service by the application software. See the application note at the end of this section.

SYNCHRONOUS OPERATION (MODE 0)

Mode 0 is the synchronous operating Mode 0 of the Serial I/O Port. It is intended primarily for transferring data to external shift registers or for communication with serial peripheral devices. The word length is eight bits on both transmit and receive. Serial data is both input and output on the RXD pin. Both transmit and receive data are synchronized to a clock signal which is output on the TXD pin at the serial data rate fixed at 1/12 of the frequency of the clock oscillator. A block diagram of the serial I/O port and timing waveforms for Mode 0 is shown in Figure 14–2 as a reference for the following discussion.

Serial data output is initiated following any instruction which causes data to be written to the Transmit Shift register located at the SBUF register address. At the time that data is written to the Transmit Shift register, a 1 is simultaneously written to the 9th bit position of the register (D8). The internal WRSBUF signal is pulsed during S6P2 and data is shifted out LSB first beginning at S6P2 of the next machine cycle. The contents of the Transmit Shift register are shifted to the right one position during S6P2 of every machine cycle until D7 has been output. As each shift right operation is performed, a 0 is shifted into the MSB position from the left. At the end of the D7 bit time, another shift is performed at S6P2 which loads the output latch of RXD with the 1 which was originally written into bit position D8. During the final shift register operation, another 0 is shifted in from the left so that the Transmit Shift register contains all 0's. Also at this time, the Transmit Interrupt flag (TI) is set and a serial interrupt will be generated if enabled.

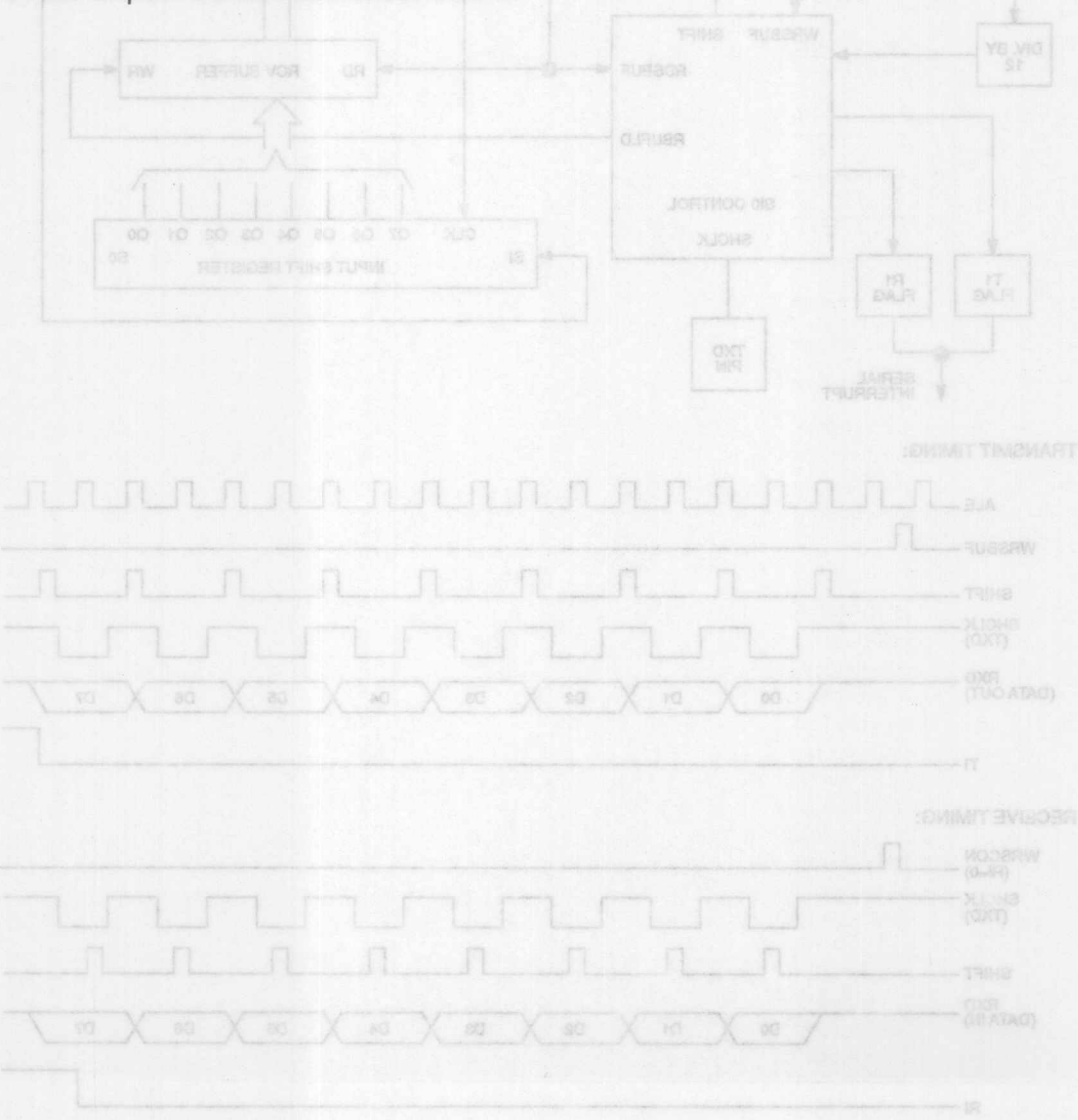
During serial data transmission in Mode 0, SHCLK is initially driven low onto the TXD pin at S3 of the machine cycle when D0 is output. During the time that the data word is shifted out, SHCLK will be low during S3, S4, and S5 and high during S6, S1, and S2 of every machine cycle.

A serial data word will be shifted into the Receive Shift register as soon as the condition REN=1 and RI=0 is satisfied. This condition can only be initiated by a write to

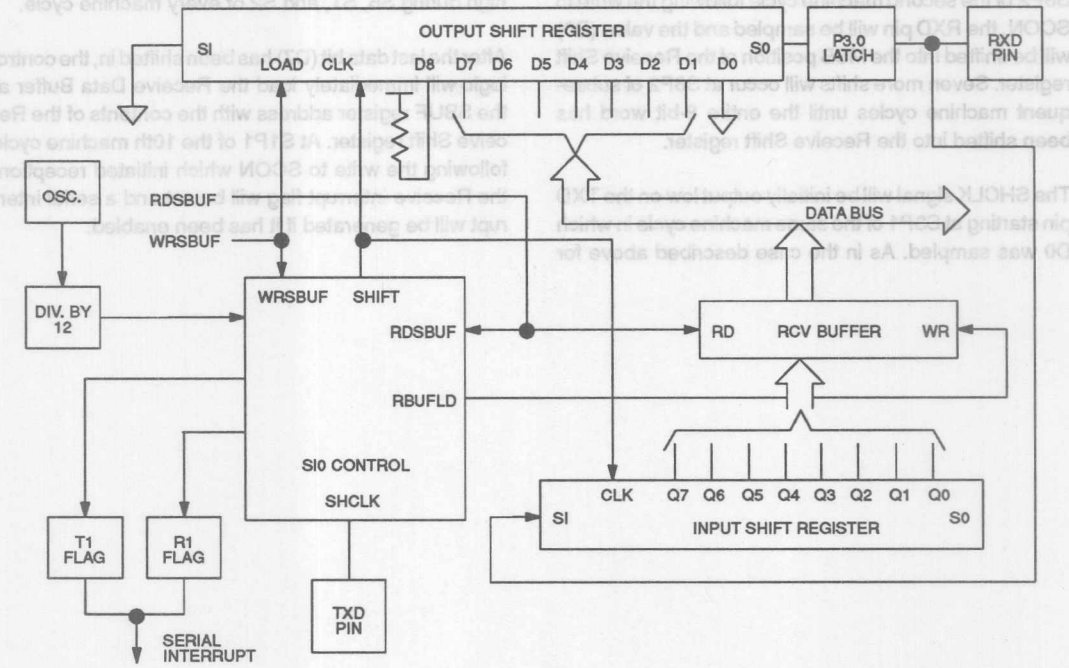
the SCON register from the application software. At S6P2 of the second machine cycle following the write to SCON, the RXD pin will be sampled and the value (D0) will be shifted into the MSB position of the Receive Shift register. Seven more shifts will occur at S6P2 of subsequent machine cycles until the entire 8-bit word has been shifted into the Receive Shift register.

The SHCLK signal will be initially output low on the TXD pin starting at S3P1 of the same machine cycle in which D0 was sampled. As in the case described above for transmit, SHCLK will be low during S3, S4, and S5 and high during S6, S1, and S2 of every machine cycle.

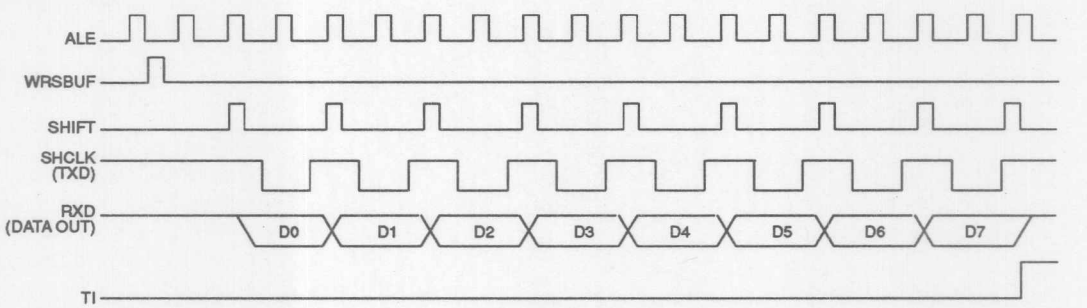
After the last data bit (D7) has been shifted in, the control logic will immediately load the Receive Data Buffer at the SBUF register address with the contents of the Receive Shift register. At S1P1 of the 10th machine cycle following the write to SCON which initiated reception, the Receive Interrupt flag will be set and a serial interrupt will be generated if it has been enabled.



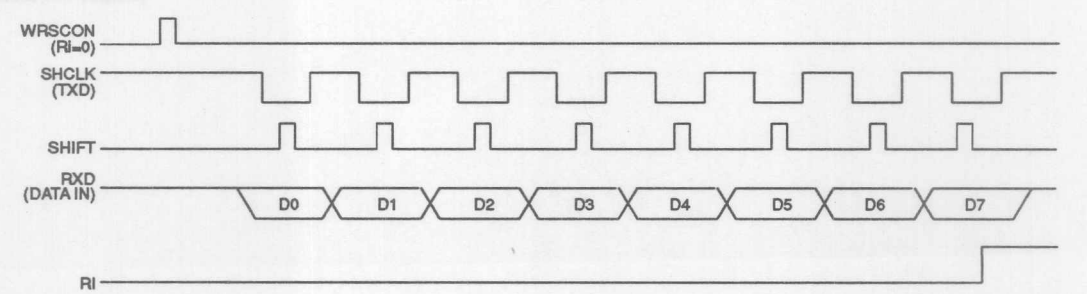
MODE 0 BLOCK DIAGRAM AND TIMING Figure 14–2



TRANSMIT TIMING:



RECEIVE TIMING:



ASYNCHRONOUS OPERATION

Mode 1, 2, and 3 provide asynchronous, full-duplex communication via the Serial I/O Port. The serial data word is either 10 or 11 bits long, depending on the mode selected. All three modes include one start bit, eight data bits, and one stop bit. Modes 2 and 3 include an additional, programmable 9th data bit. TXD is used for serial data output, while RXD is used for serial data input. In all three modes, the serial data word is both transmitted and received LSB first. The baud rate generator clock pulse (BRG clock) is derived either from the Timer 1 overflow output or divided directly from the clock oscillator frequency (of period t_{CLK}). The following description applies to all three of the operational modes. Figure 14–3 is a functional block diagram of the operation of the serial I/O port in Mode 1 including the timing waveforms which should be referred to in the discussion below.

Asynchronous serial data output begins whenever a write operation from the application software occurs with the SBUF register location as the destination address. When the write operation occurs at the time indicated by the WRSBUF signal in the timing diagram, the contents of the 8-bit data bus will be loaded into bits D8–D1 of the Transmit Shift register. Simultaneously, a 0 will be loaded into the D0 bit position of the shift register and a 1 will be loaded into the Stop bit position. (D9 for Mode 1, D10 for Modes 2 or 3).

During data transmission, the clocking frequency provided by the output of Timer 1 is divided down by a factor of 16 by the hardware to establish the serial output bit rate. Following the write operation to the Transmit Shift register, the LSB will be shifted out to the output latch of the TXD pin at the next time the divide-by-16 counter rolls over to zero. This counter is not synchronized to the machine cycles associated with instruction execution. As a result, data transmission will commence anywhere from 0 to 16 of the Baud Rate Generator clocks from the time that the Transmit Shift register is written. Successive bits from the Transmit Shift register will be shifted into the output latch of the TXD pin each time the divide-by-16 counter rolls over to zero. As each shift right operation is performed, a 0 is shifted into the MSB position from the left. When the Stop bit is shifted into the latch, the shifting operation is complete and the TI flag will be set. A serial interrupt will be generated if it has been enabled.

The Baud Rate Generator clock output is fed directly into the Bit Detector to perform serial data reception. Reception begins when a valid start bit o 0 is detected on the RXD pin. The Bit Detector will determine when this has occurred as follows: On each BRG clock pulse, the RXD pin will be sampled for a 1-to-0 transition. When such a transition is recognized, the Big Detector will then reset its own internal divide-by-16 counter and sample the RXD pin on the 7th, 8th, and 9th BRG clock times following the transition. If a logic 0 level is detected on two out of these three sample times, a valid start bit is assumed. Otherwise, the Bit Detector will reject the incoming signal as a start bit and will repeat the process by searching for another 1-to-0 transition on RXD.

If a valid start bit is detected, the RXD pin will be sampled in the middle of each successive bit time until the entire 10-bit or 11-bit serial word has been received. Following the detection of a valid start bit, successive bit times begin each time that the Bit Detector's divide-by-16 counter rolls over to 0. During each bit time, the RXD pin is sampled on the 7th, 8th, and 9th BRG clock times. For the data bits, the logic level which is read at least two out of the three sample times by the Bit Detector will be the one which is shifted into the Receive Shift register. Just after the logic level is detected during the 10th bit time, the control logic will test to see if the following conditions are true:

a) The previous sate of RI was 0.

b) SM2=0; or if SM2=1, then if the 10th received bit=1.

If these conditions are met during the 10th bit time, then the control logic will not perform another shift, but will instead load the contents of the Receive Shift register into the Receive Data Buffer, load the logic state determined at the Stop bit time into the RB8 status flag (if SM2=0), and set the RI bit. A serial interrupt will then be generated if it has been enabled via the IE register. If the above conditions are not satisfied during the stop bit time, then the received word is lost.

The first condition is interpreted by the control logic to mean an "overrun" condition has been detected. This means that a serial data word has been received before the application software read the last received word from the Receive Data Buffer. Except for a hardware reset, the condition RI=1 resulting from a previously received word can only be cleared by a write from the application software. It is therefore intended to be cleared by the application software after the Receive Data Buffer contents has been read from the SBUF register address. This signals the hardware that a properly received data word has been processed by the application

software. In an overrun condition with RI=1, the originally received word will remain in the Receive Data Buffer and all successively received data words will be lost.

When SM2=1, received data words will be selectively discarded in a manner depending on the asynchronous mode selected.

The operational details which are unique to each of the asynchronous modes are summarized below.

Mode 1

In Mode 1, the asynchronous serial data word is ten bits long, including one start bit, eight data bits, and one stop bit. The baud rate generation is derived from the Timer 1 overflow output and is therefore programmable. Figure 14–3 is a functional block diagram of the operation of the serial I/O port in Mode 1 operation including the timing waveform.

In Mode 1 operation, the SM2 bit may be used to discard a received serial data word in which a "framing error" is detected, i.e., when a valid stop bit has not been detected. When SM2=1. the incoming serial data word will be ignored unless the received Stop bit=1. If SM2=0, then the value of the received Stop bit will be loaded into the RB8 status flag so that it may be processed by the application software.

Mode 2 and 3

In Mode 2 and 3, the asynchronous serial data word is 11 bits long, including one start bit, eight data bits, a programmable 9th data bit, and one stop bit. For Mode 2, the Baud Rate Generator clock is programmable to either 1/32 or 1/64 of the clock oscillator frequency (f_{CLK}), depending on the state of the SMOD bit (PCON.7) as follows:

SMOD	BRG CLOCK
0	$\frac{f_{CLK}}{64}$
1	$\frac{f_{CLK}}{32}$

For Mode operation, the baud rate generator clock is the Timer 1 Overflow output as described for Mode 1.

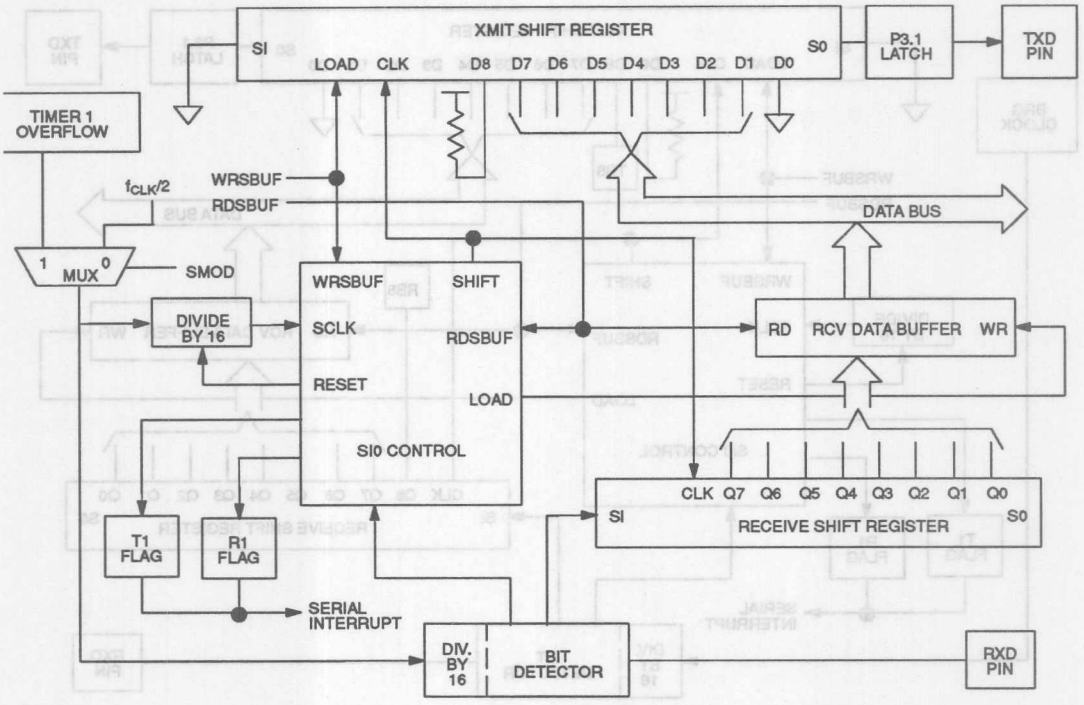
Transmission and reception takes place for Modes 2 and 3 as described except as noted below.

When the Transmit Shift register is written in Mode 2, the register is simultaneously written with a 0 in bit position D0 for a Start bit and a 1 is written into D10 for a Stop bit. D9 is the programmable bit which is written with the state of TB8 (SCON.3). TB8 can be written with the value of 1 or 0 by the application software.

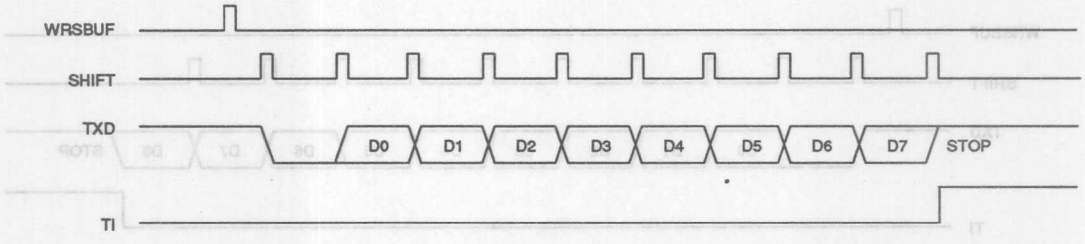
On receive, the eight data bits are shifted into the Receive Shift register following the detection of a valid Start bit. After the Stop bit has been detected, the Receive Data Buffer will be loaded with the contents of the Receive Shift register if RI=0 and SM2=0. Also at this time, the programmable 9th data bit will be loaded into RB8 in the SCON register. If RI=1 after the time the Stop bit is sample, then the incoming word will be lost.

The SM2 flag may be used in the implementation of a multiple processor communication scheme by selectively discarding incoming serial data words according to the state of the programmable 9th data bit. When SM2=1, only those words in which this 9th bit is a 1 will be loaded into the Receive Data Buffer and cause a serial interrupt to be generated. Thus, the programmable 9th bit can be used to flag an incoming data character as an address field as opposed to a data field for example.

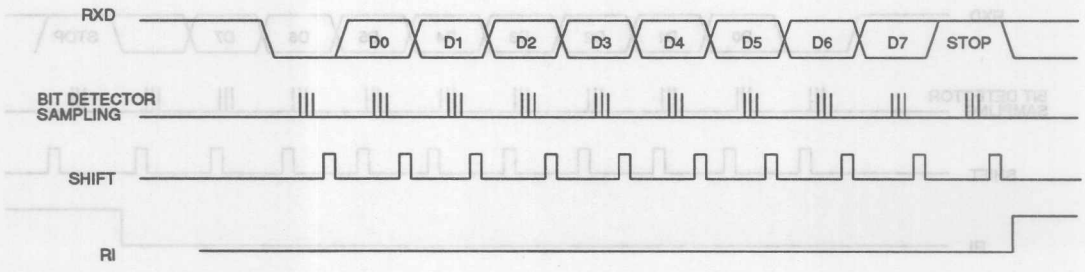
SERIAL PORT MODE 1 BLOCK DIAGRAM Figure 14–3



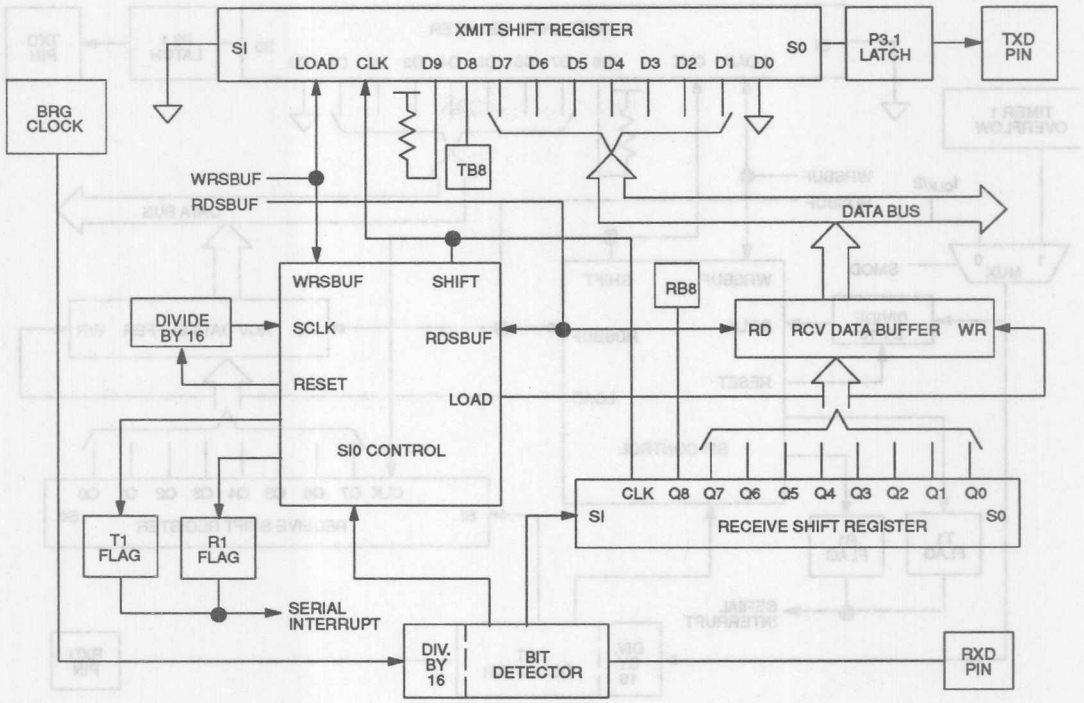
TRANSMIT TIMING:



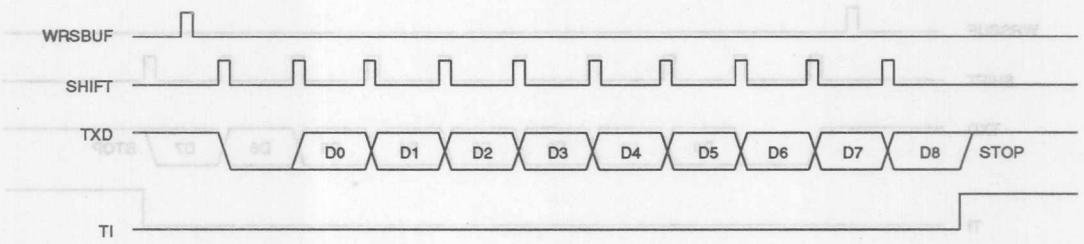
RECEIVE TIMING:



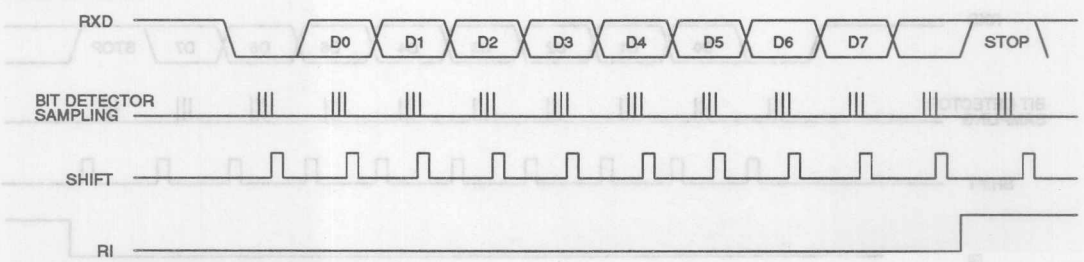
MODE2 AND 3 BLOCK DIAGRAM Figure 14–4



TRANSMIT TIMING:



RECEIVE TIMING:



APPLICATION: SERIAL PORT SETUP

The Soft Microcontroller UART can provide either synchronous or asynchronous serial communication. This note demonstrates the UART setup including the reasons for and the software to perform particular setups. This example shows asynchronous communication with a PC COM port.

A typical goal of Soft Micro to PC communication is to transfer stored data from the nonvolatile RAM. This example will show how to move 256 bytes from NVRAM to the PC via the serial port. Once the 256 bytes have been received by the PC, it will send confirmation. For this example, the confirmation code will be A5h. the Soft Micro will be assumed to run at 11.0592 MHz, a common crystal choice. The purpose of this example is mainly to illustrate the Serial Port setup. The first decision is the communication rate. The most common choice is 9600 bits per second. The Soft Micro and the PC are capable of communicating at up to 57,600 bps. This example will demonstrate both 9600 and 19,200 bps. A typical application has some form of error checking built into the data, so no parity is required. This code will therefore run at 8N1 or 8 bits, no parity, 1 stop bit. This is a common selection for PC terminal emulator software. Thus the setup summary is as follows:

Communication type :	Asynchronous
Baud Rate :	9600, 19200
Bits per word :	8
Stop bits :	1

As shown in the following table, this most closely corresponds to Serial Mode 1.

SERIAL I/O OPERATING MODES

MODE	SYNC/ASYNC	BAUD CLOCK	DATA BITS	START/STOP	9TH DATA BIT FUNCTION
MODE 0	SYNC	12 t_{CLK}	8	None	None
MODE 1	ASYNC	Timer 1 Overflow	8	1 Start 1 Stop	None
MODE 2	ASYNC	32 t_{CLK} or 64 t_{CLK}	9	1 Start 1 Stop	0, 1, or parity
MODE 3	ASYNC	Timer 1 Overflow	9	1 Start 1 Stop	0, 1, or parity

The Serial Port is controlled by the SCON register. Serial Interrupts will also be used. These are controlled by IE and IP. The setup for each SFR is shown below. In addition, Mode 1 is associated with Timer 1, which is controlled by TCON and TMOD.

Mode 1 is selected using the SCON register. The table from the SCON register shown below indicates that Mode 1 is selected by choosing the value SM0 = 0 and SM1 = 1.

SMO	SM1	MODE	FUNCTION	WORD LENGTH	BAUD CLOCK
0	0	Mode 0	Synchronous	8 bits	12 t_{CLK}
0	*1*	*Mode 1*	*Asynchronous*	*10 bits*	*Timer 1 Overflow*
1	0	Mode 2	Asynchronous	11 bits	32 or 64 t_{CLK}s
1	1	Mode 3	Asynchronous	11 bits	Timer 1 Overflow

SM0 = 0 and SM1 = 1 corresponds to the value SCON.7 = 0 and SCON.6 = 1. In addition the since the application requires receiving data, the serial receiver must be enabled. This is done by setting the REN bit at SCON.4 to a logic 1. The remaining bits in SCON can be written to 0. Thus the value for SCON is 01010000b or 50h.

SCON – 98h			
SM0	SM1	SM2	REN
0	1	0	1

TB8	RB8	TI	RI
0	0	0	0

This application uses the serial interrupt. It serves two purposes. First, the software knows when byte has been sent, so it knows when another can be written. Second, after the 256 bytes have been transmitted, the PC will respond. It is not know when this will occur and the software may have other tasks to attend. Therefore the serial interrupt will inform the micro when the confirmation code has been received by the UART. This example will enable only the serial Interrupt. It will be set for high priority since in a real system, other interrupts might be enabled.

IE – 0A8h			
EA	–	–	ES
1	0	0	1

ET1	EX1	ET0	EX0
0	0	0	0

To enable interrupts, the EA bit must be set. In addition, the setting the ES bit turns on the Serial Interrupt. Thus the value 10010000b or 90h will enable serial interrupts. Note that although a timer will ultimately be used to generate serial baud rates, the timer interrupt is not used. As mentioned above, this example will use a high priority for the Serial Interrupt. This is done as follows.

IP – 0B8h			
RWT	–	–	PS
0	0	0	1

PT1	PX1	PT0	PX0
0	0	0	0

The Serial Port Interrupt has been set to a high priority by setting the PS bit to a logic 1. Thus the Serial Port is configured for high priority by writing a 00010000b or 10h to the IP register at location 0B8h.

The Serial Port is now configured. The only remaining task is to set the correct baud rate. The example stated above that the communication rate would be either 9600 or 19,200 baud. To generate baud rates in Serial Mode 1, the Timer 1 is used. The UART uses the Timer 1 overflow, then divides this frequency by either 16 or 32 to generate the internal baud rate clock. Each time the Timer 1 value increments past 0FFh is considered an overflow. Due to the formula used for generating baud rates shown below, the 11.0592 MHz crystal assumed for this example generates good baud rate values. This is the most convenient and commonly used choice for generating baud rates. Other convenient values are 7.3728 MHz and 1.8432 MHz.

To get 9600 bits per second, the baud rate generator must create an interval af 1/9600 seconds = 1.0416 ms. To get 19,200 bits per second, the interval is 1/19200 = 520.83 µs. Note that the timers count up, so the value that the timer starts from must be selected to generate the desired interval. The following values are useful:

1/11.0592 MHz = $t_{CLK} = 90.4 * 10^{-9}$

Timer runs at 12 t_{CLK} per count = $1.085 * 10^{-6}$

Time out = (256 – Timer start value) * 12 t_{CLK} = (256 – Timer start value) * $1.085 * 10^{-6}$

UART Baud rate clock = 1/baud rate = (16 or 32) * Time out

Whether 16 or 32 is used in the baud rate generator is determined by the SMOD bit at PCON.7. 16 is used for SMOD=1, and 32 is used for SMOD=0. This is commonly referred to by the expression $(2^{SMOD})/32$. Since SMOD is either 0 or 1, this value is either 1/32 or 2/32 respectively.

The user selects the time–out value and the setting for SMOD to set the baud rate. This is done as follows.

$$\text{Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{1}{12t_{CLK} * (256 - TH1)}$$

This formula solves as :

$$TH1 = 256 - \frac{2^{SMOD}}{32 * 12\ t_{CLK} * BaudRate}$$

For 9600 = Baud rate, TH1 = FDh with SMOD = 0.

To create 19,200 baud, the SMOD bit should be set to a logic 1 with the same value for TH1. SMOD has the effect of doubling the baud rate for any time out value.

TH1 – 8Dh

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	1	1	1	0	1

PCON – 87h

SMOD	$\overline{POR}$	PFW	WTR	EPFW	EWT	STOP	IDL
0	X	X	X	X	X	X	X

TMOD – 89h

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0
0	0	1	0	X	X	X	X

As shown in the TCON description, setting M1 = 1 and M0 = 0 selects Timer 1 mode 2 which is the 8–bit auto–reload mode. In this example, Timer 0 is not used, so the lower four bits of TMOD are unused. Therefore the TMOD register can be written with 00100000b or 20h.

TCON – 88h

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
0	1	0	0	0	0	0	0

In summary the following SFRs are configured to enable the serial port for 9600 baud asynchronous operation:

TH1	FDh
SCON	50h
IE	90h
IP	10h
TMOD	20h
TCON	40h
PCON	00h (SMOD=0)

To set up the Serial Port for 19,200 baud, the only difference is that the SMOD bit at PCON.7 is set. Therefore, writing 80h to PCON will accomplish this.

The value for TH1 and SMOD have been determined. The only remaining task is to configure the Timer 1 for 8–bit auto reload operation. This will cause the timer to start counting from the TH1 value after each time out. The TMOD register is set as follows:

The remaining step is to enable the timer. Once this is done, the baud rate generator will be in operation and serial I/O can be performed. The TCON register is used to enable the timers as shown below. The TR1 bit is set to a logic 1 to enable the Timer 1 function. Writing a 01000000b or 40h to TCON will do this.

The software that configures the serial port can be simply seven move instruction the configure the SFRs mentioned above to the value as shown. This example will show this code in the context of performing the application described at the beginning of this note.

The application example described moving 256 bytes of data from memory to the serial port, then receiving a confirmation code of 0A5h. The memory will be assumed to be located in the MOVX RAM beginning at the Partition address. For this example, the Partition will be 4000h and the micro will be a DS5000. Using a DS5001 would change the value written to the MCON to configure the Partition.

```
TA              Equ     0C7h
MCON            Equ     0C6h
Org             00h

Reset :
SJMP            Start

Org             23h

Serial_ISR :
CLR             RI              ;Clear receive flag
CLR             TI              ;Clear transmit flag
RETI                            ;No special processing, return to application

Org             30h

Start :
MOV             TA, #0AAh       ;Start Timed Access
MOV             TA, #55h        ; finish Timed Access
CLR             RI              ;Initialize receive flag
CLR             TI              :Initialize transmit flag
MOV             MCON, #88h      ;Select Partition at 4000h
MOV             SCON, #50h      ;Configure Serial Port for Mode 1 and enable receiver
MOV             TMOD, #20h      ;Configure the Timer 1 for 8-bit auto-reload
MOV             TCON, #40h      ;Enable the Timer 1
MOV             TH1, #0FDh      ;Set Baud Rate
ANL             PCON, #7Fh      ;Set SMOD=0
MOV             IP, #10h        ;Set Serial Interrupt to high priority
MOV             IE, #90h        ;Globally enable interrupts and Serial Interrupt

Send :
MOV             R0, #0FFh       ;Set loop counter to 255d
MOV             DPTR, #4000h    ;Put the data pointer at the beginning of data memory

Send_Loop :
MOVX            A, @DPTR        ;Get data byte
MOV             SBUF, A         ;Transmit data byte
JNB             TI, $           ;Wait for serial word to be sent (interrupt)
INC             DPTR            ;Next byte to be sent
DJNZ            R0, Send_Loop   ;Decrement loop counter

Receive :
JNB             RI, $           ;Wait for incoming word (interrupt)
MOV             R1, SBUF        ;Get received byte
CJNE            R1, #0A5h, Send ;Check for confirmation code
                                ; and resend all data if wrong

End
```

SECTION 15: CPU TIMING

OSCILLATOR

The Soft Micro provides an on-chip oscillator circuit which may be driven either by using an external crystal as a time base or from a TTL-compatible clock signal. The oscillator circuitry provides the internal clocking signals to the on-chip CPU and I/O circuitry.

The schematic shown in Figure 15–1 illustrates the required connections when using the Soft Micro with a crystal. Typically, the values of C1 and C2 should both be 33 pF. If a resonator is used, C1 and C2 should be 47 pF.

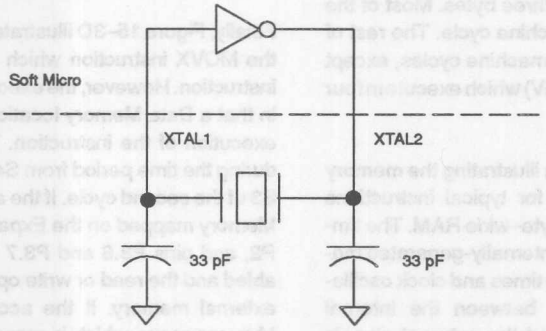
XTAL1

Input to the inverting oscillator amplifier and input to the internal clock generating circuits.

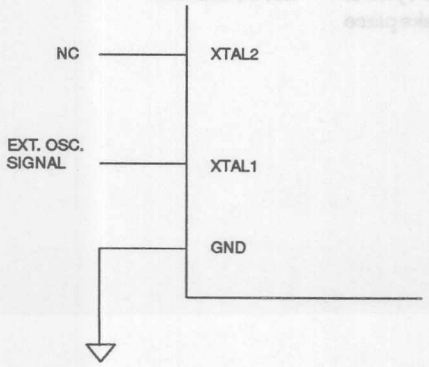
XTAL2

Output from the inverting oscillator amplifier. This pin is also used to distribute the clock to other devices.

CRYSTAL CONNECTION Figure 15–1



CLOCK SOURCE INPUT Figure 15–2



Oscillator Characteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator as shown in Figure 15–1. The crystal should be parallel resonant, AT cut type.

To drive the device from an external clock source, XTAL1 should be driven, while XTAL2 is left unconnected as shown in Figure 15–2. There are no requirements on the duty cycle of the external clock signal since the input to the internal clocking circuitry is through a divide-by-two flip-flop. However, minimum and maximum high and low times specified in the electrical specifications must be met to insure proper operation.

INSTRUCTION TIMING

The internal clocking signals are divided down to produce the necessary clock phases, state times, and machine cycles which define the sequential execution of instructions. Two clock oscillator periods define one state time. The first clock oscillator pulse period of a state time is called the Phase 1 clock. while the second is called the Phase 2 clock. In general, arithmetic and logical operations take place during Phase 1 and internal register-to-register transfers take place during Phase 2.

A machine-cycle is composed of a total of twelve oscillator periods or six state times. The state times within the machine cycle are numbered S1 through S6. Each clock oscillator period within the machine cycle is designated according to the state number and the phase it represents within the state. Thus, the oscillator periods are numbered S1P1 (State 1, Phase 1) through S6P2 (State 6, Phase 2).

All of the instruction sequences executed by the CPU are preceded by a single byte (8-bit) opcode and consist of a total of either one, two, or three bytes. Most of the instructions execute in one machine cycle. The rest of the instructions execute in two machine cycles, except for multiply (MUL) and divide (DIV) which execute in four cycles each.

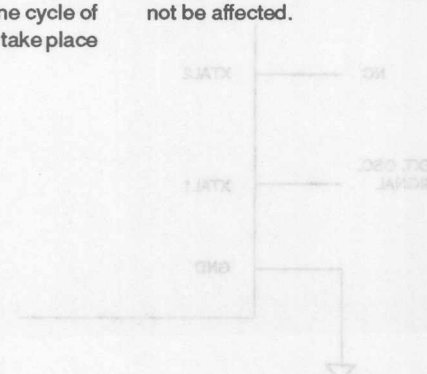
Figure 15–3 is a timing diagram illustrating the memory access and execution timing for typical instructions when they are executed from Byte–wide RAM. The timing shown is referenced to the internally-generated machine cycles composed of state times and clock oscillator phases. The relationship between the internal instruction execution timing and the external signals XTAL2 and ALE is illustrated in the diagram. Except for the MOVX instructions, two code bytes from Program Memory are always read during each machine cycle of instruction execution. These read operations take place at state times S1 and S4.

Execution of a 1-byte, 1-cycle instruction is illustrated in Figure 15–3A. It begins with the opcode byte fetch which occurs during S1 and the opcode byte is latched into the Instruction register at S1P2. The code byte which is read during S4, in this case, is actually the opcode byte of the next instruction. This byte is effectively discarded and the Program Counter is not incremented. Execution of the instruction is completed S6P2, the end of the machine cycle.

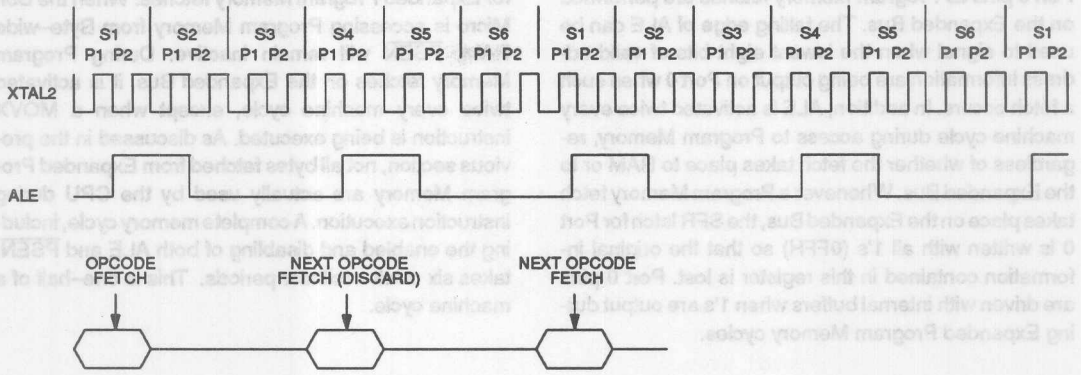
In the 2-byte 1-cycle instruction shown in Figure 15–3B, the opcode is read during S1 while the second byte of the instruction, or the operand, is read during S4. Again, execution of the instruction is complete at the end of S6P2.

A 1-byte, 2-cycle instruction is shown in Figure 15–3C. In this case the opcode byte is read at S1 of the first machine cycle. The next opcode is then read three times during the S1 and S4 of the second machine cycle. The information is discarded each of these times until it is finally read when the next instruction is actually executed.

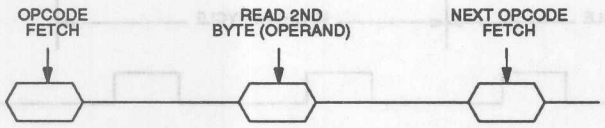
Finally, Figure 15–3D illustrates the execution of one of the MOVX instruction which is also a 1-byte, 2-cycle instruction. However, the execution timing of this unique in that a Data Memory location is accessed during the execution of the instruction. This access takes place during the time period from S4 of the first cycle through S3 of the second cycle. If the access is made from Data Memory mapped on the Expanded Bus, then ports P0, P2, and pins P3.6 and P3.7 will automatically be enabled and the read or write operation will take place on external memory. If the access is made from Data Memory space which is mapped within the Byte–wide RAM, then the read or write operation will take place on the Byte–wide RAM bus and the external port pins will not be affected.



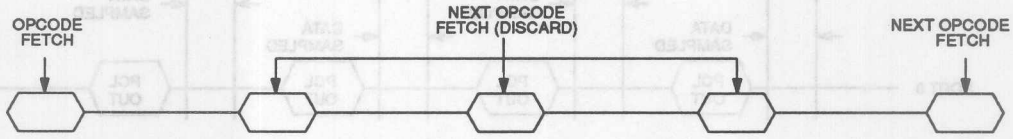
BYTE–WIDE RAM INSTRUCTION EXECUTION TIMING Figure 15–3



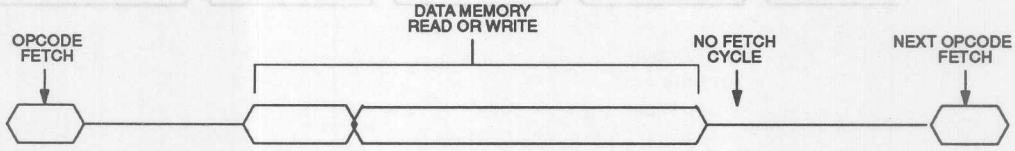
A) 1–BYTE, 1–CYCLE INSTRUCTION (E.G., DEC A)



B) 2–BYTE, 1–CYCLE INSTRUCTION (E.G., MOV A, #DATA)



C) 1–BYTE, 2–CYCLE INSTRUCTION (E.G., INC DPTR)



D) MOVX: 1–BYTE, 2–CYCLE INSTRUCTION

EXPANDED PROGRAM MEMORY TIMING

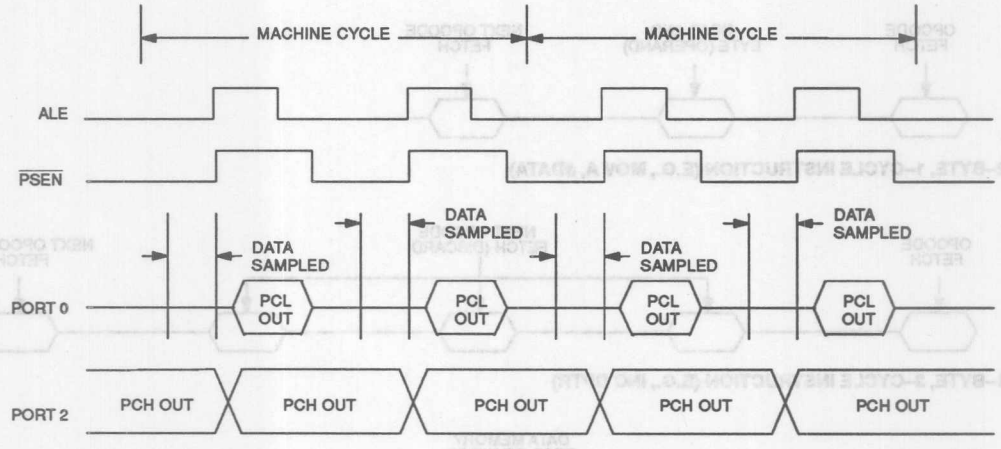
A Program Memory access will occur on the Expanded Bus any time that instructions are executed from Program Memory space which is mapped outside of the Byte–wide RAM. Mapping of Program Memory on the Expanded Bus is dependent on the programming of the Partition, Range, the state of the external $\overline{EA}$ pin, and the internal Security Lock. Refer to Section 4 for a detailed discussion on Program Memory mapping.

The external timing for the Expanded Program Memory fetch cycle is illustrated in Figure 15–4. A full 16-bit address is always output on the multiplexed Expanded Bus (P2, P0) pins whenever such an access is performed. The high-order eight bits will be output on the P2 pins while the low-order eight bits will be output on the P0 pins. Strong pull-ups are enabled onto Ports 0 and 2 for the duration of time that 1's are output on the port for address bits. As long as Program Memory is being executed from the Expanded Bus, P0 and P2 pins are unavailable for use as general-purpose I/O.

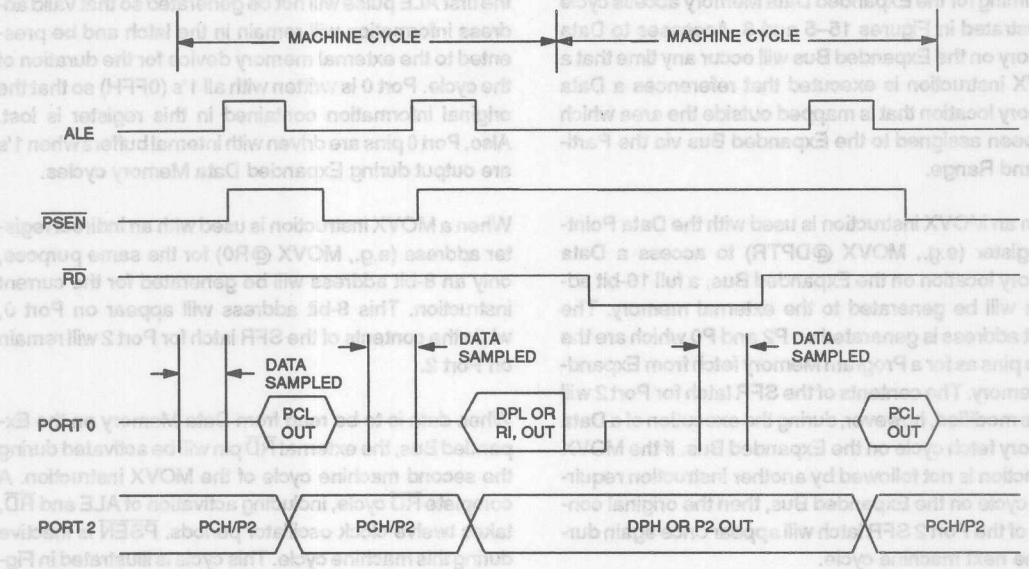
Multiplexed address and data information appear on the Port 0 pins as Program Memory fetches are performed on the Expanded Bus. The falling edge of ALE can be used to signal when the lowest eight bits of valid address information are being output on Port 0 when such a fetch occurs. In addition, ALE is activated twice every machine cycle during access to Program Memory, regardless of whether the fetch takes place to RAM or to the Expanded Bus. Whenever a Program Memory fetch takes place on the Expanded Bus, the SFR latch for Port 0 is written with all 1's (0FFH) so that the original information contained in this register is lost. Port 0 pins are driven with internal buffers when 1's are output during Expanded Program Memory cycles.

The $\overline{\text{PSEN}}$ signal is provided as the read strobe pulse for Expanded Program Memory fetches. When the Soft Micro is accessing Program Memory from Byte–wide RAM, $\overline{\text{PSEN}}$ will remain inactive. During Program Memory fetches on the Expanded Bus, it is activated twice every machine cycle, except when a MOVX instruction is being executed. As discussed in the previous section, not all bytes fetched from Expanded Program Memory are actually used by the CPU during instruction execution. A complete memory cycle, including the enabled and disabling of both ALE and $\overline{\text{PSEN}}$, takes six clock oscillator periods. This is one–half of a machine cycle.

EXPANDED PROGRAM MEMORY FETCH Figure 15–4

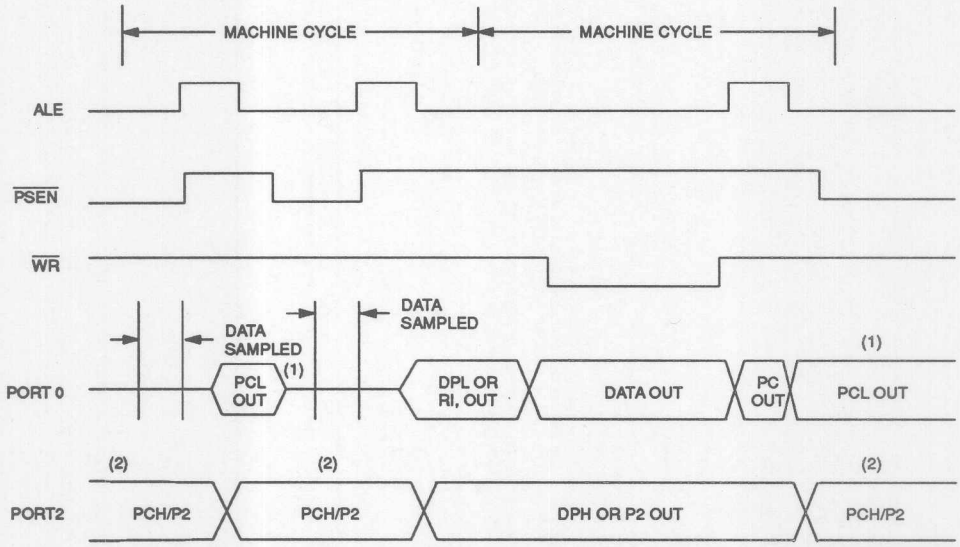


EXPANDED DATA MEMORY READ Figure 15–5



* PCL OUT if program memory also on Expanded Bus – float if not.
**PCH OUT if program memory also on Expanded Bus.

EXPANDED DATA MEMORY WRITE Figure 15–6



(1) PCL OUT if program memory also on Expanded Bus – float if not.
(2) PCH OUT if program memory also on Expanded Bus.

EXPANDED DATA MEMORY TIMING

The timing for the Expanded Data Memory access cycle is illustrated in Figures 15–5 and 6. Accesses to Data Memory on the Expanded Bus will occur any time that a MOVX instruction is executed that references a Data Memory location that is mapped outside the area which has been assigned to the Expanded Bus via the Partition and Range.

When an MOVX instruction is used with the Data Pointer register (e.g., MOVX @DPTR) to access a Data Memory location on the Expanded Bus, a full 16-bit address will be generated to the external memory. The 16-bit address is generated on P2 and P0 which are the same pins as for a Program Memory fetch from Expanded Memory. The contents of the SFR latch for Port 2 will not be modified, however, during the execution of a Data Memory fetch cycle on the Expanded Bus. If the MOVX instruction is not followed by another instruction requiring a cycle on the Expanded Bus, then the original contents of the Port 2 SFR latch will appear once again during the next machine cycle.

Multiplexed address/data information is output on Port 0 during the execution of a Data Memory cycle on the Expanded Bus. The falling edge of ALE can be used to latch the lower eight bits of address information into an external transparent latch (e.g., 74LS373 or equivalent). During the second cycle of a MOVX instruction, the first ALE pulse will not be generated so that valid address information will remain in the latch and be presented to the external memory device for the duration of the cycle. Port 0 is written with all 1's (0FFH) so that the original information contained in this register is lost. Also, Port 0 pins are driven with internal buffers when 1's are output during Expanded Data Memory cycles.

When a MOVX instruction is used with an indirect register address (e.g., MOVX @R0) for the same purpose, only an 8-bit address will be generated for the current instruction. This 8-bit address will appear on Port 0, while the contents of the SFR latch for Port 2 will remain on Port 2.

When data is to be read from Data Memory on the Expanded Bus, the external $\overline{RD}$ pin will be activated during the second machine cycle of the MOVX instruction. A complete $\overline{RD}$ cycle, including activation of ALE and $\overline{RD}$, takes twelve clock oscillator periods. $\overline{PSEN}$ is inactive during this machine cycle. This cycle is illustrated in Figure 15–5. When the MOVX instruction specifies a write operation to the external memory device, the $\overline{WR}$ signal will be activated as shown in Figure 15–6. Data is output on Port 0 just before $\overline{WR}$ is activated and remains valid until it goes back to its inactive level at the conclusion of the cycle.

SECTION 16: PROGRAM LOADING

INTRODUCTION

Program loading is performed to initialize the contents of NVRAM and to configure the Soft Microcontroller. Loading is done using a Bootstrap ROM Loader built into all Soft Micros. When this Bootstrap Loader is invoked, the user's NVRAM appears as data memory to the ROM and can therefore be initialized. Once loading is complete, the Bootstrap ROM then becomes transparent. It has no effect on the user's memory map and is completely invisible. Bootstrap Loading is normally done for the initial program loading. There are no restrictions on using it for upgrades or reprogramming, though it is possible using the memory mapping features, to perform partial reloads without invoking the loader.

The Bootstrap Loader is generally used to initialize memory. However, it is capable of several other functions. It can change the memory map configuration, dump or verify the contents of memory, perform a CRC check, fill a block with a constant, manipulate the security features, and manipulate the I/O ports. It can not display or edit the Scratchpad RAM (128 bytes) or SFRs since it uses these resources for its own operation. Note that the Scratchpad RAM will be erased by the loader. Therefore the loader should not be invoked while critical data is stored in this area. The MOVX RAM area will not be altered by the loader unless a user requests that this be done.

Each version of Soft Micro has different loader modes and different commands. All versions are capable of being programmed via the serial port and this is the preferred method. The DS5000 series is also capable of a parallel programming technique similar to an EPROM based 8051. The DS5001 series has alternate parallel load mode that is for use by a host microprocessor using the 8042–type RPC mode. The following discussions explain the methods of invoking the Bootstrap Loader. Then each mode is described. The remaining sections give a detailed description of the commands and syntax used by the Bootstrap ROM.

The DS5000 series [DS5000(T), DS2250(T), and DS5000FP] and the DS5001 series [DS5001FP, DS2251(T), DS5002FP, DS2251(T)] have different program loading modes available. The DS5000 series can be loaded via its serial port or in a parallel fashion like an EPROM type 87C51. The serial mode allows the DS5000 to be programmed in a fixture or while installed in the end system. The parallel method requires a super–voltage and is normally done in a fixture only. The DS5001 series has a similar serial mode with the same benefits. The parallel mode is entirely different. The DS5001, using its RPC slave interface, can be loaded in a parallel manner by a host microprocessor. This is also an in–system technique but could be performed in a fixture. It requires no super–voltage pulses. Note that this mode is a high–speed loader and bares no resemblance to an 87C51 load mode.

Note: Dallas Semiconductor highly recommends that serial load capability be designed into the target system. This provides substantial flexibility to upgrade and troubleshoot the system. Using in–system serial loading allows a product to take full advantage of the Soft Micro's features.

INVOKING THE BOOTSTRAP LOADER

The Soft Micro will ordinarily power–up or come out of reset in its normal operating mode. It requires external hardware to cause the Bootstrap Loader to be invoked. This mode can be invoked at any time. Once the Bootstrap Loading session is complete, the micro will perform a reset and begin the user's code at address 0000h. This will be no different from a hardware reset, except that the Bootstrap loader had write–access to all functions. Note that the contents of the Scratchpad RAM (128 bytes) will be erased. However, the Bootstrap Loader cannot make accidental changes. As mentioned above, the hardware aspects are different from the DS5000 series and the DS5001 series. They are described individually below.

DS5000 Series

The DS5000 is placed in its Program Load configuration by simultaneously applying a logic 1 to the RST pin and forcing the $\overline{\text{PSEN}}$ line to a logic 0 level. Immediately following this action, the DS5000 will look for a serial ASCII carriage return (0DH) character received at 57600, 19200, 9600, 2400, 1200, or 300 bps over the serial port or a Parallel Program Load pulse. For whichever type is first detected, the DS5000 will place itself in the associated Program Load mode. If an ASCII <CR> character is detected first, then the DS5000 will place itself in the Serial Program Load mode and will ignore any Parallel Program Strobe pulses. Conversely, if a Parallel Program Strobe pulse is first detected, then the DS5000 will be placed in the Parallel Program Load mode and all incoming data on the serial port will be ignored. The selected program load mode will remain in

PSEN=1) is removed.

In normal programming of the DS5000, problems with selection of the incorrect Program Load mode will not be encountered. When the DS5000 is placed in an 8751-compatible programming system, no serial data will be applied on the RXD pin for the serial port. As a result, there is almost no chance of random activity on this pin being interpreted as a <CR> character at a valid baud rate. Similarly, serial program loading will most often be performed in the end system. Consequently, there is again almost no chance of a valid Parallel Program Strobe pulse (with V_{PP} voltage at 13 V) being interpreted as a signal to invoke the Parallel Program Load mode when the Serial Program Load mode is desired.

If the Program Load configuration is removed such that RST=0 and $\overline{PSEN}$=1 with power still applied at V_{CC}, the device will undergo an internal hardware reset and will begin executing code from the reset vector at 0000h in Program Memory.

DS5001 Series

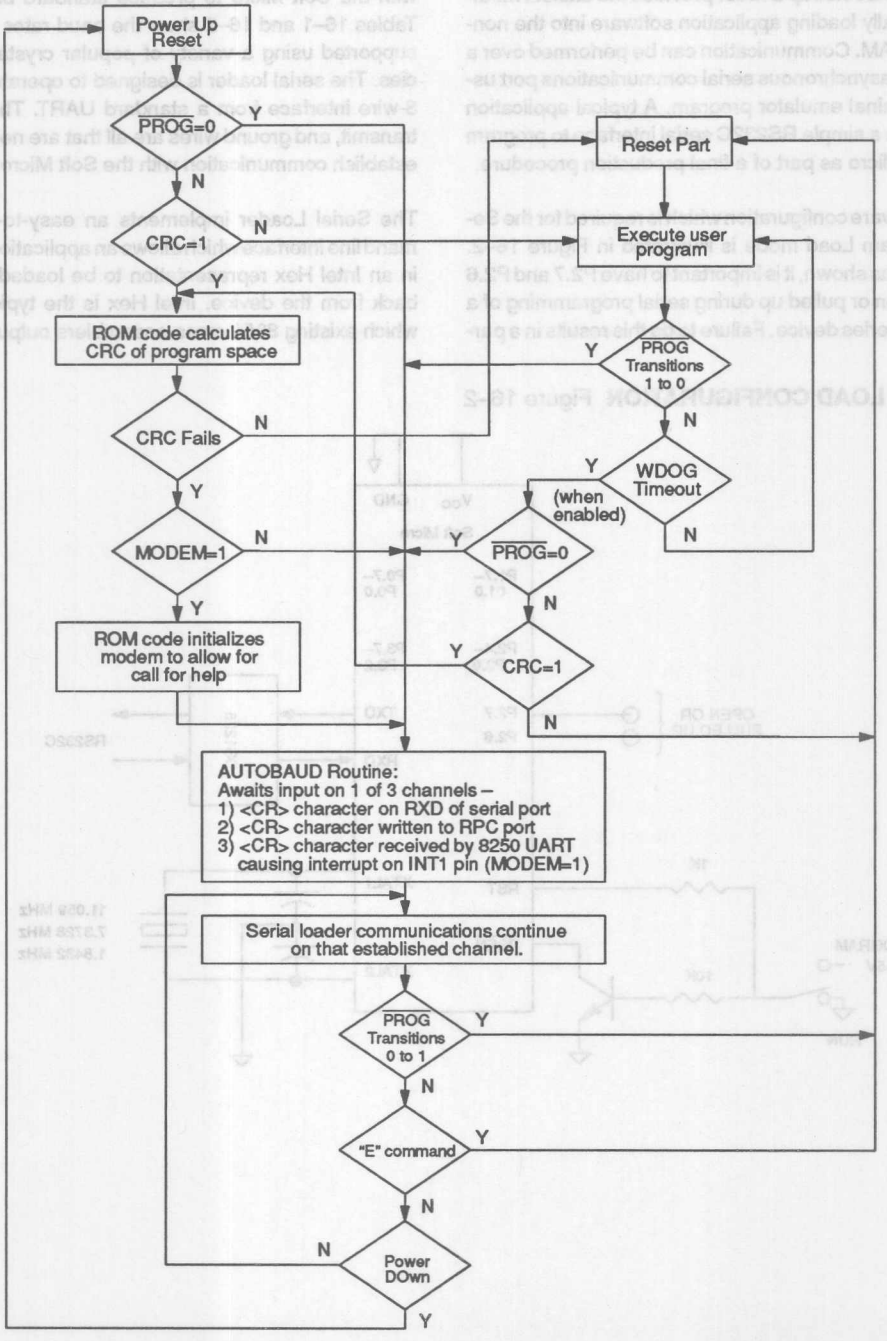
The DS5001 series Bootstrap Loader can be invoked with a single program pin. A falling edge on the $\overline{PROG}$ pin will invoke the loader. Note that the $\overline{PROG}$ pin must remain low for 48 oscillator clocks to be certain of recognition. Taking the $\overline{PROG}$ pin to a logic 1 will remove the loader and cause the DS5001 to perform a reset. Note also that the $\overline{PROG}$ pin must be high for as much as 48 clocks before the CPU is guaranteed to exit the loader. This constitutes the "pseudo–edge" detection of the pro-

Once the loader mode stimulus has been detected, the DS5001 will begin looking for an ASCII carriage return (0Dh). It will look at the serial port and the RPC (8042) port. For serial reception, the loader will auto–baud at 57600, 19200, 9600, 2400, 1200 and 300 bps. For RPC mode, the 0Dh value must be written into the Data In buffer as described in the RPC section under Parallel I/O. When either of these conditions is detected, the loader will place itself in that loader mode. Activity on the other port will be ignored. This condition will remain until the loader is exited or power is cycled. When the loader stimulus is removed, the processor will perform a hardware reset and begin execution at location 0000h.

EXITING THE LOADER

The normal method of leaving the loader is to remove the stimulus that invoked it. In the DS5000 series, the RST pin, must be driven low or allowed to float and the $\overline{PSEN}$ signal should be allowed to float. The RST pin has an internal pull–down. The $\overline{PSEN}$ is an output and will drive itself. Note that both of these conditions must occur or the loader will not be exited. For the DS5001, there are several options. If the RST and $\overline{PSEN}$ option is used, the they must be removed as described above. If $\overline{PROG}$ is pulled low, it can either be returned high or the Exit "E" command can be issued. Since the loader is edge activated, this will restart the user's code even while the $\overline{PROG}$ pin is low. If power were to cycle while the $\overline{PROG}$ pin were low, the loader would be invoked on power up. The flow of these conditions is shown in Figure 16–1.

INVOKING AND EXITING THE LOADER ON THE DS5001 SERIES Figure 16–1



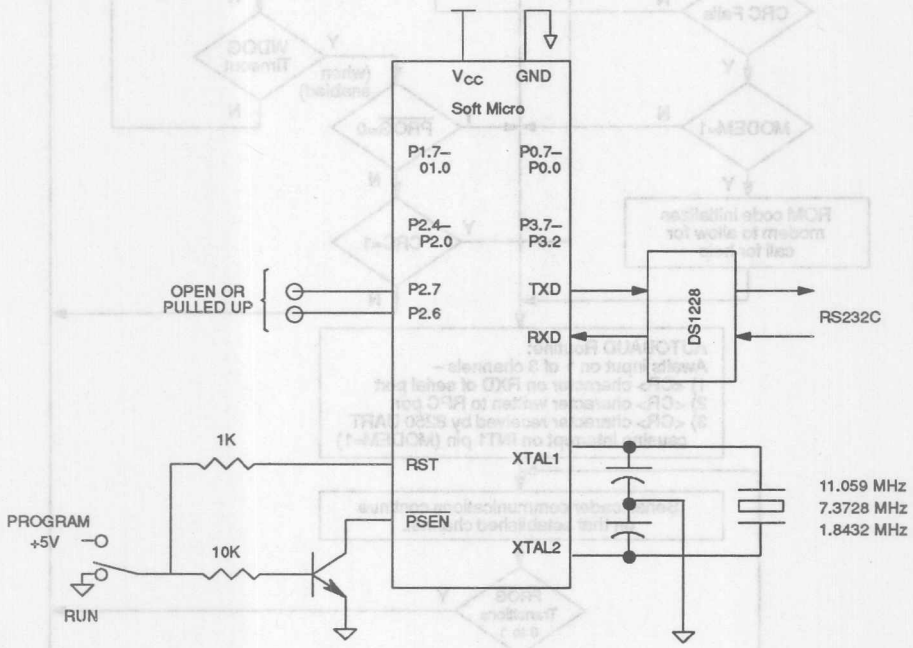
SERIAL PROGRAM LOAD MODE

The Serial Bootstrap Loader provides the easiest method of initially loading application software into the nonvolatile RAM. Communication can be performed over a standard asynchronous serial communications port using a terminal emulator program. A typical application would use a simple RS232C serial interface to program the Soft Micro as part of a final production procedure.

The hardware configuration which is required for the Serial Program Load mode is illustrated in Figure 16–2. Note that as shown, it is important to have P2.7 and P2.6 either open or pulled up during serial programming of a DS5000 series device. Failure to do this results in a parallel load operation. A variety of crystals can be used with the Soft Micro to produce standard baud rates. Tables 16–1 and 16–2 show the baud rates which are supported using a variety of popular crystal frequencies. The serial loader is designed to operate across a 3-wire interface from a standard UART. The receive, transmit, and ground wires are all that are necessary to establish communication with the Soft Micro.

The Serial Loader implements an easy-to-use command line interface which allows an application program in an Intel Hex representation to be loaded and read back from the device. Intel Hex is the typical format which existing 8051 cross-assemblers output.

SERIAL LOAD CONFIGURATION Figure 16–2



AUTO-BAUD RATE DETECTION

The Serial Bootstrap Loader has the capability of determining which of the six supported baud rate frequencies is being used for communication and initializing its internal hardware for communication at that frequency. When the Program Load mode is first invoked, the device will watch for activity on the serial port. If a <CR> character is received at one of the six supported baud rates, then operation in the Serial Program Load mode will be established. The loader expects to talk asynchronously at 300, 1200, 2400, 9600, 19200, or 57600 baud using eight data bits, no parity, and one stop bit in full duplex. A break signal followed by a carriage return will cause a re-determination of baud rate. Although an 11.0592 MHz crystal is standard for generating baud rates, the auto-baud rate detector allows a variety of crystals to be used. If a crystal frequency other than 11.0592 MHz is used, then the baud rate frequencies which will be recognized by the serial loader are shown in Table 16–1. Table 16–2 shows baud rates and crystals that have an error of ±3%. This is usually sufficient for communication. The user should verify this in the end system.

SERIAL LOADER BAUD RATES FOR DIFFERENT CRYSTAL FREQUENCIES Table 16–1

CRYSTAL FREQ (MHz)	BAUD RATE					
	300	1200	2400	9600	19200	57600
14.7456		Y	Y	Y	Y	
11.0592	Y	Y	Y	Y	Y	Y
9.21600	Y	Y	Y	Y		
7.37280	Y	Y	Y	Y		
5.52960	Y	Y	Y	Y		
1.84320	Y	Y	Y	Y		

SERIAL LOADER BAUD RATES FOR DIFFERENT CRYSTAL FREQUENCIES Table 16–2

CRYSTAL FREQ (MHz)	BAUD RATE (±3%)				
	300	1200	2400	9600	19200
16.000000		Y	Y		
15.000000		Y	Y	Y	Y
14.745600		Y	Y	Y	Y
14.318180		Y	Y	Y	Y
12.000000		Y	Y		
11.059200	Y	Y	Y	Y	Y
11.000000	Y	Y	Y	Y	Y
10.000000		Y	Y		
9.216000	Y	Y	Y	Y	
8.000000		Y			
7.372800	Y	Y	Y	Y	Y
6.144000	Y	Y	Y		
6.000000	Y	Y	Y		
5.990400	Y	Y	Y		
5.120000	Y	Y	Y		
5.068800	Y	Y	Y		
5.000000	Y	Y	Y		
4.915200	Y	Y	Y		
4.608000	Y	Y	Y		
4.433620	Y	Y	Y		
4.194300	Y				
4.096000	Y				
4.032000	Y				
3.579545	Y	Y	Y	Y	Y
2.457600	Y	Y			
2.000000	Y				
1.843200	Y	Y	Y	Y	

Y indicates that the baud rate for that particular crystal is supported by serial loader auto–baud rate detection scheme with an error of ±3%.

SERIAL BOOTSTRAP LOADER INITIALIZATION

When an incoming <CR> character is detected at a valid baud rate, the Serial Bootstrap Loader will be initialized and will transmit the following message:

```
DS500x SERIAL LOADER Vn.n

>
```

The revision number of the firmware is displayed in place of n.n above. The ">" is the prompt character which is displayed to indicate that the Serial Bootstrap Loader is ready to receive a command.

COMMAND LINE INTERFACE

The Soft Micro implements an easy-to-use command line interface which is very similar to those found in debugging environments. The Serial Bootstrap Loader responds to single character alphabetic commands which are summarized below. There are differences between versions as noted. A detailed description of each command follows.

COMMAND	FUNCTION	VERSION
C	CRC–16 of RAM	All
D	Dump Intel hex file	All
E	Exit loader	DS5001 series
F	Fill RAM with a constant	All
G	Get value from ports	All (DS5000 after Rev. D4)
I	Include CRC	DS5001 series
K	Load 40–bit key	DS5000 series
L	Load Intel Hex file	All
M	Modem	DS5001 series
N	New – invoke Freshness	DS5001 series
P	Put a value to the ports	All (DS5000 after Rev. D4)
R	Read configuration	All
T	Trace (echo) incoming data	All
U	Unlock security	All
V	Verify RAM against incoming Hex	All
W	Write configuration	All
Z	Lock	All
^C	Reset loader	All
Xon/Xoff	Flow control of serial transmission	All

Selected commands require arguments and some commands have optional arguments. In all cases, arguments are expected to be hexadecimal numbers. In addition, an ASCII control-C character (^C) will cause the Serial Loader to terminate any function currently being executed and display the command line prompt. An incoming break character (defined as a received null character (00H) with the stop bit = 0) will cause the Serial Bootstrap Loader to be restarted and the baud rate redetermined.

COMMAND LINE SYNTAX

Single-letter ASCII characters are recognized as commands. Arguments are represented by hexadecimal numbers. A hexadecimal number is any sequence of hexadecimal characters. A hexadecimal character may be a digit, 0 through 9, or one of the letters A through F. A byte will always be the right-most two digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following bytes:

A → 0AH
AB → 0ABH
ABC → 0BCH
ABCD → 0CDH

An address will always be the right-most four digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following addresses:

A → 000AH
AB → 00ABH
ABC → 0ABCH
ABCD → 0ABCDH
ABCDE → 0BCDEH

The D and F commands allow optional addresses to be entered. The syntax [Begin-Address [End-Address]] is used to convey the following meanings:

a) No arguments: Begin-Address is set to 0 and End-Address is set to the Range.

b) One argument: Begin-Address is set to the argument and End-Address is set to the Range.

c) Two arguments: Begin-Address is set to the first argument and End-Address is set to the second argument. This second address may exceed the address value specified by the Range.

In cases b and c, the End Address may not be less than the Begin Address, either implicitly or explicitly. RAM will be addressed from 0 to 1FFF for 8K RAM and from 0 to 7FFF for 32K RAM. This maximum value is determined by the Range.

Error messages will be printed as soon as errors are detected. All messages are preceded by the two characters 'E:', and followed by a mnemonic description.

Commands will not be processed until an entire command line is entered and terminated with a <CR>. No command line may be greater than 16 bytes, which is the maximum number of characters in the K command. Since a command line is not processed until a <CR> is entered, it may be edited with the delete key which will do a destructive delete to the screen. Lines longer than 16 characters will cause an error message to be displayed and no action to be taken on the command line.

Only legal characters will be echoed back to the screen. The legal characters are: 0123456789 <:>, <space>, ABCDEFGHIJKLMNOPQRSTUVWXYZ, and <delete>. Backspace characters (<BS>) are converted to delete characters. The horizontal tab character is converted to space. Lower case alphabetic characters are converted to upper case alphabetic.

The <delete> character is executed as a <BS> <space> <BS> when possible in command mode. This will cause the character to be overprinted on a hardcopy device. The <CR> character generates a <CR> <LF> pair.

The Serial Bootstrap Loader will respond to XOFF characters by stopping transmission as soon as the character is received. A control-C or an XON will resume serial transmission. The Serial Bootstrap Loader will not transmit a XOFF character. The program is able to keep up with input as long as the receiver can keep up with its output. The receiver should be programmed to quit transmission after it sends an XOFF and transmit anything before sending an XON.

Intel Hex data is not echoed unless the Trace mode is toggled on.

COMMAND SUMMARIES

^C

Interrupt whatever is going on, clear all the buffers, put up a prompt and wait for the next command. Anything in the type-ahead buffer is removed. All output is stopped. If trace had been on before, it is cleared. If XOFF had been in effect, it is cleared.

C [begin-address [end-address]]

Return the CRC-16 (cyclic redundancy check) of the NV RAM. This computation is performed over the Range unless optional start and end addresses are given. The CRC-16 algorithm is commonly used in data communications.

D [begin-address [end-address]]

Dump memory in Intel Hex Format. An optional address range may be specified. Each record will contain up to 32 data bytes. The last line printed is the end–of–data record.

E

The serial loader is exited. This works if a negative edge on $\overline{\text{PROG}}$ was used to invoke it, or a CRC check failed.

F byte [begin-address [end-address]]
Fill memory with the value of the specified byte. An optional address range may be specified.

G
Data is read from ports 0, 1, 2 and 3 and is printed as four pairs of hexadecimal digits.

I
A CRC–16 is computed from 0 to CRC_RANGE minus 2 and the computed CRC is put into CRC_RANGE minus one and CRC_RANGE. This is used when power–on CRC checking is desired. This command will print DONE when it finishes.

K byte-1 byte-2 byte-3 byte-4 byte-5
Put a Key into the encryption key word. The five bytes that will be entered are echoed before they are put into the registers.

L
Load standard Intel Hex formatted data into memory. Only record types 00 and 01 are processed. Each line of the file is treated the same way. All characters are thrown away until a header character ':' is read. The rest of the record, as defined by the length byte, is processed. Control will return to the command prompt after an Intel End Record is encountered. Every time a byte is put to memory, it is read back to verify that it is there. If the byte read back is different, an error is reported. All errors are reported immediately after the character is received which caused the error. The program will then read characters until a colon is found and then attempt to process the data input from the command line. Note that all bytes are put to memory as they are encountered. This means that if a bad checksum is found, an error will be reported, but all the bytes on the line will have been put to memory.

M
Toggle the status of the modem available bit (MDM in the CRC register). This will display either AVAILABLE or UNAVAILABLE.

N
This command will put the part into freshness mode if the part is powered down after executing this command.

P <Port> <Value>
This writes the requested value to the requested port. Port must be 0, 1, 2, 3.

R
Read and print the current value of the MCON register for the DS5000 series. The read command displays the current values of the MCON (C6h) register, RPCTL (D8h) register, MSL bit, CALIB (DFh) register, and CRC (C1h) register for the DS5001 series. It is printed in the following manner:
MCON:XX RPCTL:XX MSL:XX CALIB:XX CRC:XX.

T
Trace by echoing the incoming Intel Hex data. This is a toggle command and will display the state after toggling. Initially the state of the toggle is OFF.

U
Clear the Security Lock. The Range is set to 32K and the partitioning is set to all program memory. Note that unlocking the Security Lock clears the encryption registers, Vector RAM, and selects $\overline{CE1}$ for the RAM. The U and Z commands are the only commands that may be executed when the chip is locked.

V
Verify current contents of memory with the Intel Hex coming in. This command does the same thing the Load command does, except that is does not write the byte to RAM; it simply compares the byte in memory to the byte in the data stream. A message is reported for each successful verify.

W byte or
W[CALIB/CRC/MCON/MSL/RPCTL]VAL
For the DS5000 series, write the byte to the MCON register. This command is used to set the Partition, the Range, and ECE2 as desired. The PAA bit is unaffected by this command. Use the Z command to set or clear the Security Lock bit. For the DS5001 series, VAL is written to the requested register. This command is discussed below in detail.

Z
Set the Security Lock. Only the U and Z commands may be given after the Security Lock is set.

^C

Restarts most operations. N cannot be restarted.

Xon/Xoff

These two characters provide flow control to the serial loader. The serial loader will never issue either character, but will respond to both. Xoff (control–S, DS3, 0x13) requests that character transmission stop. Xon (control–Q, DC1, 0x11) requests the resumption of transmission.

NOTES ON SELECTED DS5001 COMMANDS

When using the Include command 'I', certain precautions are needed, The CRCBIT (bit 0) of the CRC SFR (C1h) must be set or the error message E:NOCRCB is printed. The MSL bit must not be cleared (via loader) or the error message E:NOTCOD is printed. If the PD bit (bit 1) of the MCON SFR (C6h) is one, then CRC_RANGE must be less than or equal to the program Range, or else the error message E:BADRNG is printed.

When storing an end system, it is desirable not to lithium–back RAM or a real–time clock. The Newness command 'N' will accomplish this. When 'N' is issued, the loader will prompt with CONFIRM: after the N is entered. The user must not type FRESH without any spaces or deletes and terminated it with a carriage return to put the part into freshness. The message DID NOT CONFIRM is printed if a mistake is made while entering FRESH; otherwise the message POWER DOWN TO MAINTAIN FRESHNESS is printed. The 'N' command should be the last function that is executed. After this, the system should be powered down for storage. At this time, the V_{CCO} will be pulled low, removing power from RAM or clock. The newness command may not be executed when taking through the modem to the serial loader. If it is attempted, the error message E:ILLCMD is printed.

The DS5001 provides loader commands to assist in system checkout. These are 'G" Get and 'P' Put. Get will read the values of all four I/O ports. Put is used to write a value to a port. This allows a measure of hardware control while the DS5001 is effectively in a reset state. If a port number other than 0, 1, 2, or 3 is used, the error message E:BADREG is printed. Ports 0 and 2 may not be altered when taking to the loader through the RPC interface. The error message E:BADREG is printed if it is attempted. Bits 0 and 1 of port 3 will always be written as ones when port 3 is altered (serial port).

As the DS5001 supports various memory configurations etc., the write command has more options than the DS5000 version. Any of the following special function registers/functions can be initialized using the loader. VAL is written to the requested register. If an illegal register name is entered, the error message E:BADREG is printed.

CALIB	Keeps a calibration byte in nonvolatile storage for later use in calibrating a real–time clock. All 8 bits of the CALIBRATION register may be altered.
CRC	This register selects the range over which a power–on CRC will be performed, and enables the process. Only bits 0, 4, 5, 6 and 7 of the CRC register may be altered. Bits 1, 2, and 3 are left alone.
MCON	Controls range, partition, and peripheral selects, All but bit 0 of the MCON register may be altered.
MSL	MSL only uses the low order bit of VAL to change the MSL bit. MSL allows the data space in a fixed memory (non–partitionable) system to be loaded using the loader or application software. In fixed partition and 128K mode, if MSL is written=0, program loads/verifies will go to data space. Upon entry, MSL will be = 1. MSL has no effect in user mode.
RPCTL	RPCTL only uses the low order bit of VAL to change bit 0 of the RPCTL register. The LSB of the RPCTL is also used to determine the Range.

Only record types 00 (data) and 01 (end) may be loaded. Other record types will cause the error message E:BADREC to be printed out, but loading will continue with the next valid record. The last two bytes of each record contain a checksum. This checksum is compared to the computed value for the record, and if different, the error message E:BADCKS is printed out. Unfortunately, the data bytes for this record will have been put to memory already. End of Data records (01) do not check for valid checksums. After a byte is put to memory, it is read back immediately to see if it is the same. If not, the error message E:MEMVER is printed.

ERROR MESSAGES

E:ARGREQ
An argument or arguments is required for this command. The F, K, and W commands require an argument.

E:BADCKS
The checksum found at the end of an Intel Hex data record is not the same as the value calculated.

E:BADCMD
An unrecognized command was entered. If the part is locked, the only commands that will be recognized are the U and Z commands. More than 16 characters entered on the command line will also cause this error.

E:BADREC
When reading an Intel Hex data record, the record type was not a 0 or 1.

E:EXTARG
Extra data was found on the command line when it wasn't necessary.

E:ILLOPT
The optional parameters given were in error. If the start address is greater than the given address, either implicitly or explicitly, then an error is printed. The range bit implicitly determines the maximum range.

E:NOTHEX
When looking for hexadecimal characters, a non-hex character was found. This error can occur when reading Intel Hex data files if any character between the colon and end of data line has a non-hexadecimal character.

E:VERIFY
The byte which was sent does not verify with the corresponding one in RAM.

INTEL HEX FILE FORMAT
8051-compatible assemblers produce an absolute output file in Intel Hex format. These files are composed of a series of records. Records in an Intel Hex file have the following format:

<Header><Hex Information><Record Terminator>

The specific record elements are detailed as follows:

Where:

:	Indicates a record beginning
ll	Indicates the record length
aaaa	Indicates the 16 bit load address
tt	Indicates the record type
dd	Indicates hex data
xx	Indicates the checksum = (2's complement (ll+aa+a+tt+dd+dd+...dd)

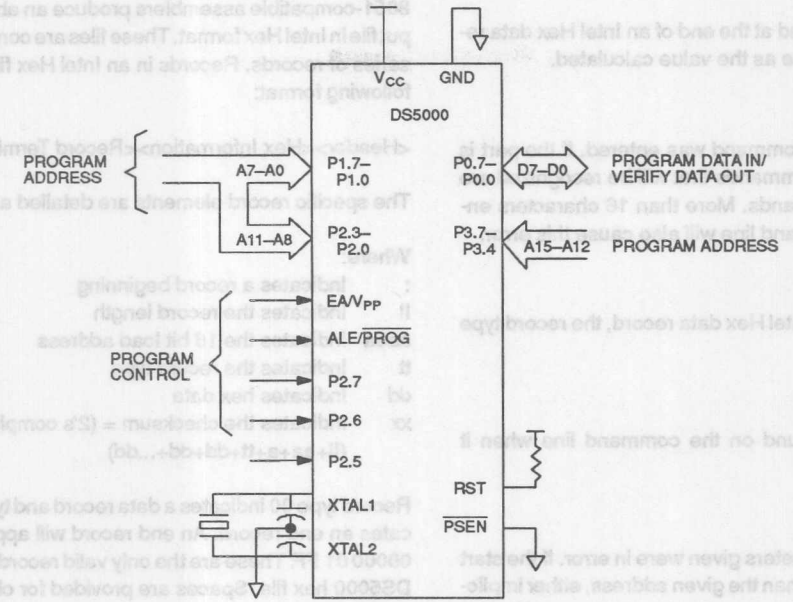
Record type 00 indicates a data record and type 01 indicates an end record. An end record will appear as :00 00000 01 FF. These are the only valid record types for a DS5000 hex file. Spaces are provided for clarity.

The following is a short Intel hex file. The data bytes begin at 01 and count up to 2F. Notice the records length, beginning address, and record type at the start of each line and the checksum at the end.

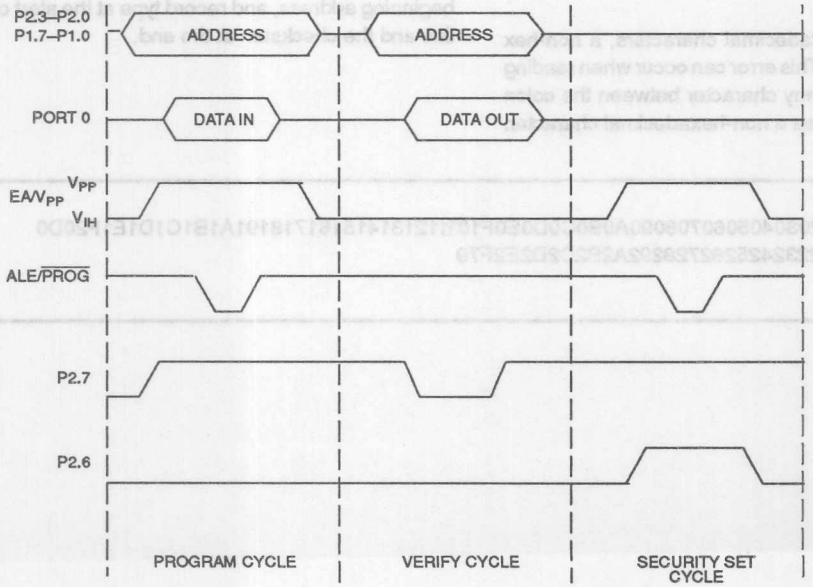
```
:20000000010203040506070809 0A0B0C0D0E0F10111213141516171819 1A1B1C1D1E1F20D0
:0F00200021222324252627282 92A2B2C2D2E2F79
:00000001FF
```

with the Program mode of the 87C51. The hardware

PARALLEL PROGRAM LOAD CONFIGURATION Figure 16–3



PARALLEL PROGRAM LOAD CYCLES Figure 16–4



PARALLEL PROGRAM LOAD MODE

Table 16–3 summarizes the selection of the available Parallel Program Load cycles. Figure 16–4 illustrates the timing associated with these cycles.

8751-COMPATIBLE PROGRAM LOAD CYCLES Table 16–3

MODE	RST	$\overline{PSEN}$	$\overline{PROG}$	$\overline{EA}$	P2.7	P2.6	P2.5
Program	1	0	0	V_{PP}	1	0	X
Security Set	1	0	0	V_{PP}	1	1	X
Verify	1	X	X	1	0	0	X
Prog Expanded	1	0	0	V_{PP}	0	1	0
Verify Expanded	1	0	1	1	0	1	0
Prog MCON or Key	1	0	0	V_{PP}	0	1	1
Verify MCON	1	0	1	1	0	1	1

The Program cycle is used to load a byte of data into a register or memory location within the DS5000. The Verify cycle is used to read this byte back for comparison with the originally loaded value to verify proper loading. The Security Set cycle may be used to enable the Software Security feature of the DS5000. One may also enter bytes for the MCON register or the Encryption Key using the Program MCON cycle. When using this cycle, the absolute register address must be presented at Port 1 and 2 as is the normal Program cycle (Port 2 should be 00H). The MCON contents can be likewise verified using the Verify MCON cycle.

When the DS5000 first detects a Parallel Program Strobe pulse or a Security Set Strobe pulse while in the Program Load mode following a Power-On Reset, the internal hardware of the DS5000 is initialized so that an existing 4 Kbyte 8751 program can be programmed into a DS5000 with little or no modification. This initialization automatically sets the Range Address for 8 Kbytes and maps the lower 4 Kbyte bank of Embedded RAM as Program Memory. The top 4 Kbytes of Embedded RAM are mapped as Data Memory. In order to program code (and thereby use the DS5000-enhanced capability), the Program/Verify Expanded cycles can be used. Up to 32 Kbytes of program code can be entered and verified. Note that the expanded 32 Kbyte Program/Verify cycles take much longer than the standard cycles.

A typical parallel loading session would follow this procedure. First, set the contents of the MCON register with the correct range and partition (if using expanded programming). Next, the Encryption Key can be loaded if desired. Then, program the DS5000 using either standard or expanded program cycles and verify. Last, turn on the security lock using a Security Set cycle.

The Security Set strobe pulse from an 8751-compatible programming system can be used to enable the Software Security feature of the DS5000. To explain this operation on the DS5000, it is useful to review how this function works with the 8751. The Security Set Strobe pulse is used to program the EPROM Security Lock bit on an 8751. The programmed bit disables the on-chip EPROM memory from being read back during a Verify cycle. The bit can only be erased by UV light when the rest of the program is erased.

With the DS5000, the Security Set Strobe pulse serves a similar function for its NVRAM-based Security Lock which when set disables the NVRAM from beginning read either through a Verify cycle in the Parallel Program mode or back through the serial port in the Serial Port mode. When a Security Set Strobe pulse is received by the DS5000, the current state of the Security Lock bit is checked. If it is currently a 0, it will be set to a 1. The Security Lock can be cleared by clearing the LSB of the MCON register.

PARALLEL PROGRAMMING CONCERNS

Dallas Semiconductor highly recommends using the serial load mode for programming the DS5000. It has proven highly reliable and easy to use. In the event that parallel programming is still desirable to some users, several incompatibilities have been discovered in actual programming system. The following is a summary of these incompatibilities:

1. The DS5000 is a fully CMOS device. It was actually designed to be pin-compatible with the 80C51/87C51 as opposed to the 8051/8751. There is a subtle difference in the pin definition between these two devices. This has to do with the oscillator inputs, XTAL1 and XTAL2. On the CMOS devices, XTAL1 is the pin which is driven in the external drive configuration. On NMOS devices, XTAL2 is driven for the external clock configuration. This difference has no effect when a crystal is tied to the pins for an external time base. However, may programming systems use the external drive configuration in order to maintain the ability to program multiple types of devices in a single 40-pin socket. For this reason, the DS5000 will not operate correctly in a 8751-compatible socket which uses the external clock mode.
2. The 87C51 data sheet specifies a "fast" programming timing algorithm for programming the locations in its on-chip EPROM memory. This algorithm is identical to the 8751 Program mode specification except for the number and duration of ALE low pulses during a "Program Byte" state. There are 25 pulses specified, each with a low time of 90 to 110 µs following by a minimum high time of 10 µs. Since the Parallel Load mode is partially implemented using firmware resident in an internal ROM memory, the 87C51 fast programming algorithm is incompatible with the DS5000. Therefore, programming systems which implement this algorithm will not correctly program a DS5000.
3. Also since the Parallel Program mode is partially firmware based, a minimum recovery time is required between back-to-back Program Byte strobes and between a Program Byte strobe followed by a Verify strobe.
4. Many programming systems apply V_{CC} voltage when programming of the device and remove it when programming is completed. This operation is compatible with the DS5000 so long as the Power On Reset time spec (t_{POR}) is met before programming begins. Since there is no similar specification on the 8751 or on the 87C51, programming systems may very well not meet the DS5000's requirements and Program strobe pulses may not be recognized by the DS5000.
5. The DS5000 is compatible with either the 21V V_{PP} programming voltage of the 8751 or the 12V V_{PP} programming voltage of the 87C51. However, some programming systems test for the amount of current that is being drawn during programming on the V_{CC} pin and/or on the V_{PP} pin. An 8751 is specified to draw a maximum of 30 mA of I_{PP} current during programming, while an 87C51 is specified for a maximum of 50 mA. A DS5000 will draw a maximum of only 15 mA of I_{PP} current during programming. As a result, these programming systems may erroneously determine that the device is incorrectly installed in the socket.

Because of the limitations cited above, Dallas Semiconductor recommends that the Serial Bootstrap Loader be used for initial program loading of the DS5000.

RPC PROGRAM MODE OPERATION

The DS5001 series offers a high–speed programming mode with many of the benefits of the Serial Loader. Like the Serial mode, it is primarily intended as an in–system technique but can be used in a fixture. This mode uses the RPC (8042) slave interface to perform a high speed parallel load. When the $\overline{PROG}$ pin is pulled to a logic 0, the Bootstrap ROM will begin looking for an ASCII carriage return. This can come in via the serial port or the RPC port. The RPC port is accessed as shown in the section on Parallel I/O. If the RPC buffer is written with a 0Dh, this will cause the loader to respond with its banner and prompt using this same interface. An external microprocessor is assumed to have written and read these values. The RPC loader implements the same command interface and syntax as the Serial

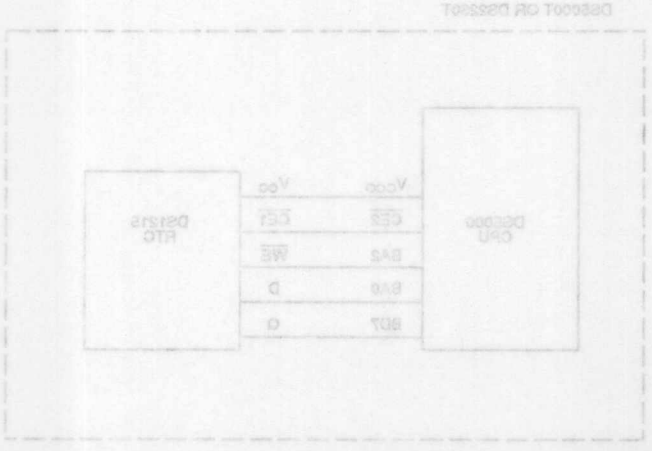
Loader. The only difference is the speed at which data can be written, and the lack of a baud rate consideration. As bytes are written into the buffer, they will be acted upon. Handshaking will be used as described in the Parallel I/O section.

The RPC mode requires no super–voltage pulses. The $\overline{CS}$, $\overline{RD}$, and $\overline{WR}$ strobes control the transfer of data between the DS5001 and the host. This protocol makes the DS5001 ideal for PC based applications, but any host processor can perform the loading.

CALL FOR HELP

The DS5001 is designed for crashproof computing. The object is to deal with all eventualities in the course of running a system. The DS5001 provides Power fail Reset, Early–warning Interrupt, and Watchdog timer as well as nonvolatile RAM for backup to achieve this. However, in the event of a problem that these features cannot deal with, the DS5001 can also Call For Help. It uses a modem, preferably the DS2245M, to accomplish this. When the DS5001 requests a call for help, the DS2245M modem performs a preprogrammed sequence of actions. This can include calling a home–base phone number and making a modem connection. The DS2245M allows a remote user to reprogram or verify the integrity of the processor. The modem aspect is discussed in the DS2245M data sheet.

As discussed above, the DS5001 is capable of performing an automatic CRC on power–up. If the CRC is incorrect, the DS5001 will invoke its Bootstrap Loader and wait for guidance. If the MDM (modem enable) bit is set in the CRC register (address C1h), the DS5001 loader will perform a Call For Help. This involves the following procedure. First, the processor will take P1.0 to a logic low. This signal should be connected to a DS2245M modem. Next, the DS5001 loader will initialize the modem's UART registers. The registers will be expected beginning at address C000h under $\overline{PE4}$. The DS5001 will then toggle the modem $\overline{OUT1}$ signal low then high via its UART registers. This causes a reset of the DS2245M modem. When the DS2245M recognizes the Call For Help pin as being low (step 1) on a reset, it performs the Call For Help procedure. This ultimately allows the remote user to examine the Bootstrap Loader directly. Notice that no user software is required to perform this function. The port at C000h will provide access to the Bootstrap Loader directly.

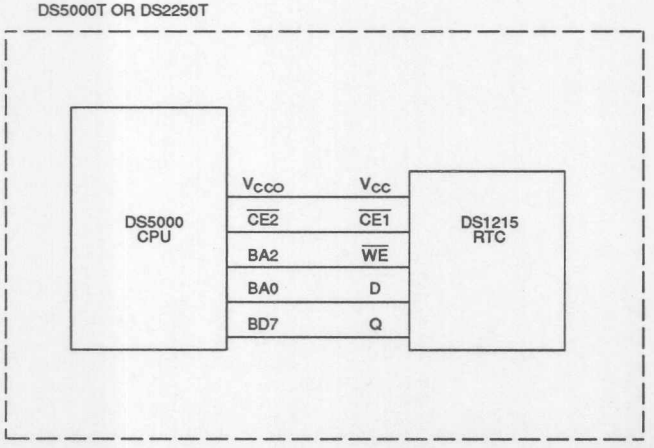


troller module versions have real–time clock (RTC) options. These include the DS5000T DIP and the DS2250T, DS2251T, and DS2252T SIMMs. In addition, user's of the monolithic microcontroller chips including DS5000FP, DS5001FP, and DS5002FP, will frequently connect a Dallas RTC. There are two types of clock used in Dallas modules. These are the Phantom type, represented by the DS1215 and the parallel type represented by the DS1283. The DS1283 is generally superior. This section provides the clock details for user's of the Time–Microcontroller modules.

DS1215 PHANTOM RTC

The first generation microcontrollers use the DS1215 Phantom RTC. These include the DS5000T and DS2250T. This clock gives permanently powered time–of–day monitoring. The clock runs from an internal 32 KHz crystal and is generally independent of the microcontroller. It provides time of day information including 0.01 second, seconds, minutes, hours, day, date, month and year. The register format is shown below. The DS1215 keeps time to 2 minutes per month accuracy. It offers a complete representation of time and calendar in a convenient BCD format. It does not provide any interrupt capability. These functions are provided in the The DS1215 is a Phantom clock because it makes no impact on the memory map. The block diagram of the DS1215 connection is shown in Figure 17–1. It is fundamentally a serial device that resides on the address bus. To access the clock, the user must set the ECE2 bit at MCON.2 to a logic 1. This will cause all MOVX instructions to access $\overline{CE2}$ instead of $\overline{CE1}$. Once ECE2 is set, the Byte–wide Address bit 2 serves as a write enable and Address bit 0 serves as the data input. Bit 7 of the Byte–wide Data bus serves as the data output. Notice that the read/write line is not used. For each $\overline{CE2}$ access, the DS1215 will watch the value of A0 on the Byte–wide bus for a particular 64–bit security pattern. This pattern checking prevents accidentally invoking the clock. Since these must be write operations, A2 must be a logic 0 for each write. The clock will take no action unless the 64 pattern bits are written in the correct order. Any error causes the pattern comparator to start over. Thus the users must "really" intend to communicate with the DS1215. Once the security pattern is written, the next 64 bits are time of day and calendar functions. Thus 128 read/writes are required for any time of day access. Data is written using BA0 and read using BD7. Thus the address actually writes data, but data is read normally using one bit.

DS5000T/DS2250T RTC BLOCK DIAGRAM Figure 17–1



The timekeeper contains a shift register with 128 locations. The first 64 locations correspond to a pattern shown in Figure 17–2. The next 64 are time data. Before access to time data may occur, the 64 bit pattern must be written. The incoming bits are checked by a pattern recognition circuit. As each correct bit of the pattern is received, the pointer is advanced. Any incorrect bit will cause the pointer to stop, and it may only be reset by a read operation. When the 64 bits of the pattern have been correctly written, access to time data begins. The next 64 bits are time data according to Figure 17–4. When the 64 bits of time data have been read or written (each bit increments the pointer), the pointer has completed its cycle of 128. The next time access is initiated by writing the pattern again. The pointer should be reset with a read operation, to make sure it is at a known location.

To write a data bit to the RTC, a MOVX instruction that forces A2 low and A0 to the state of the bit must be performed. All other address lines should be low. Address line A2 can be thought of as the write enable to the clock and A0 as the input bit. Therefore, to write the 64 bits of the pattern recognition sequence, 64 MOVX instructions must be executed. A read is performed in a similar manner, but A2 is high. Notice that data is encoded into the address line. Either a MOVX A, @DPTR or MOVX @DPTR, A will accomplish a write if the DPH contains 00H, and DPL contains 0000000Xb. The data bit is A0. The R/$\overline{W}$ signal is irrelevent.

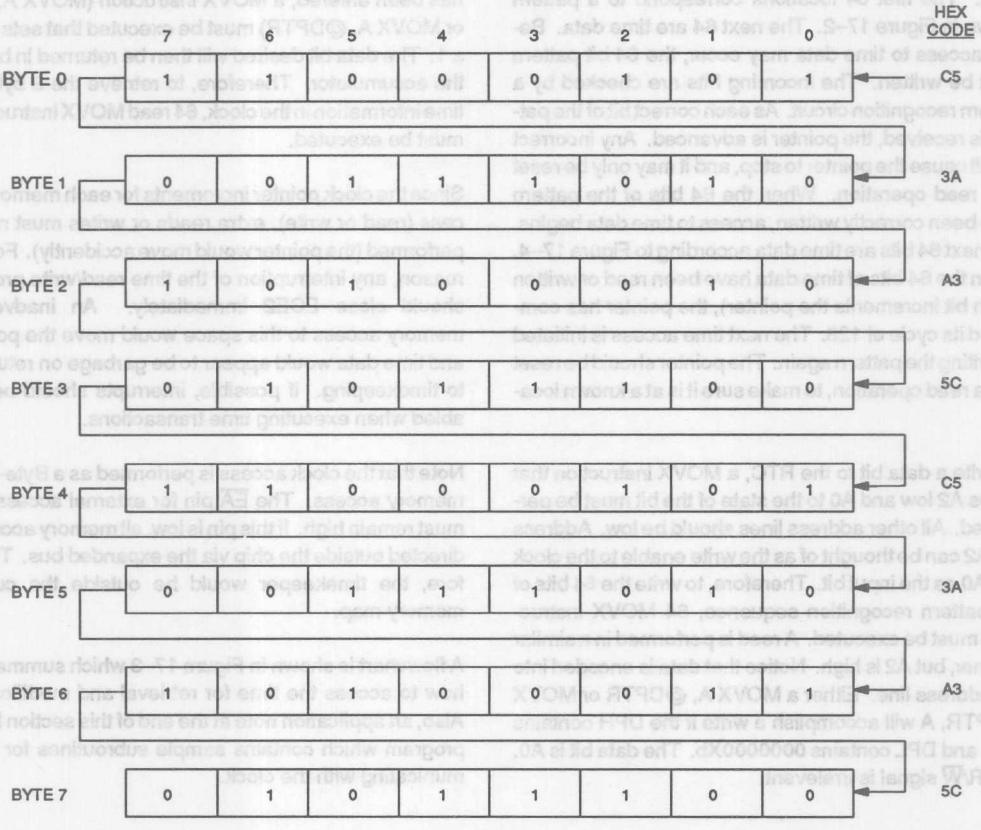
To read a data bit from the clock once the 64–bit pattern has been entered, a MOVX instruction (MOVX A, @Ri or MOVX A, @DPTR) must be executed that sets A2 to a 1. The data bit desired will then be returned in bit 7 of the accumulator. Therefore, to retrieve the 8 bytes of time information in the clock, 64 read MOVX instructions must be executed.

Since the clock pointer increments for each memory access (read or write), extra reads or writes must not be performed (the pointer would move accidently). For this reason, any interruption of the time read/write process should close ECE2 immediately. An inadvertent memory access to this space would move the pointer, and time data would appear to be garbage on returning to timekeeping. If possible, interrupts should be disabled when executing time transactions.

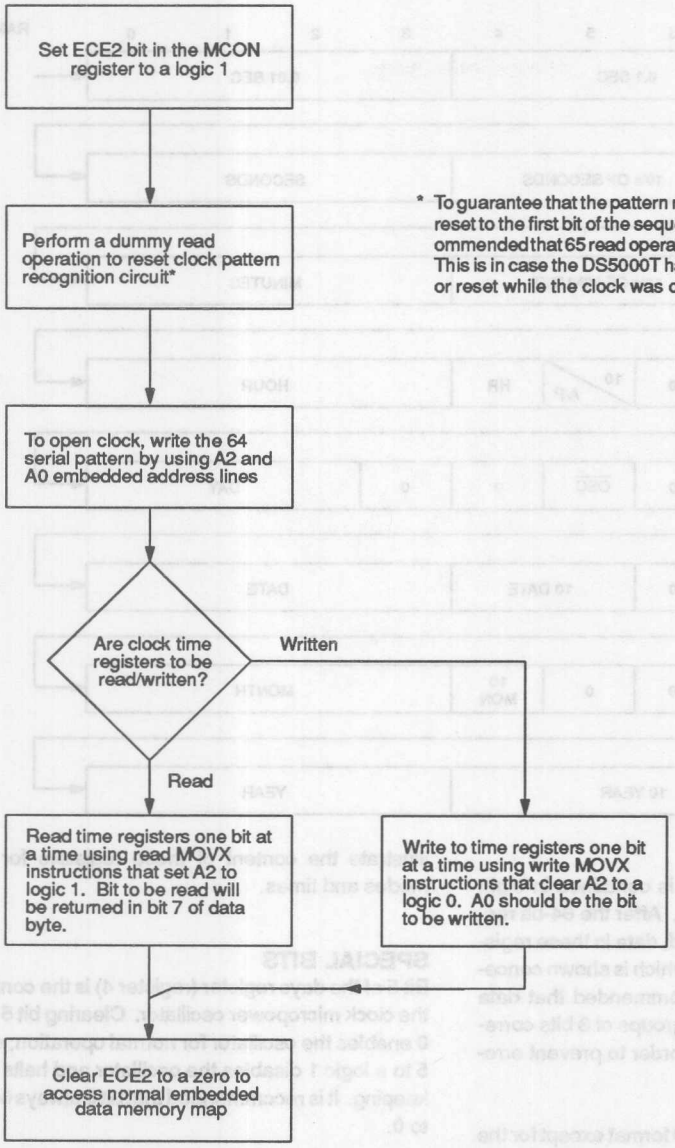
Note that the clock access is performed as a Byte–wide memory access. The $\overline{EA}$ pin for external access only must remain high. If this pin is low, all memory access is directed outside the chip via the expanded bus. Therefore, the timekeeper would be outside the current memory map.

A flowchart is shown in Figure 17–3 which summarizes how to access the time for retrieval and modification. Also, an application note at the end of this section lists a program which contains sample subroutines for communicating with the clock.

PATTERN COMPARISON REGISTER DESCRIPTION Figure 17–2

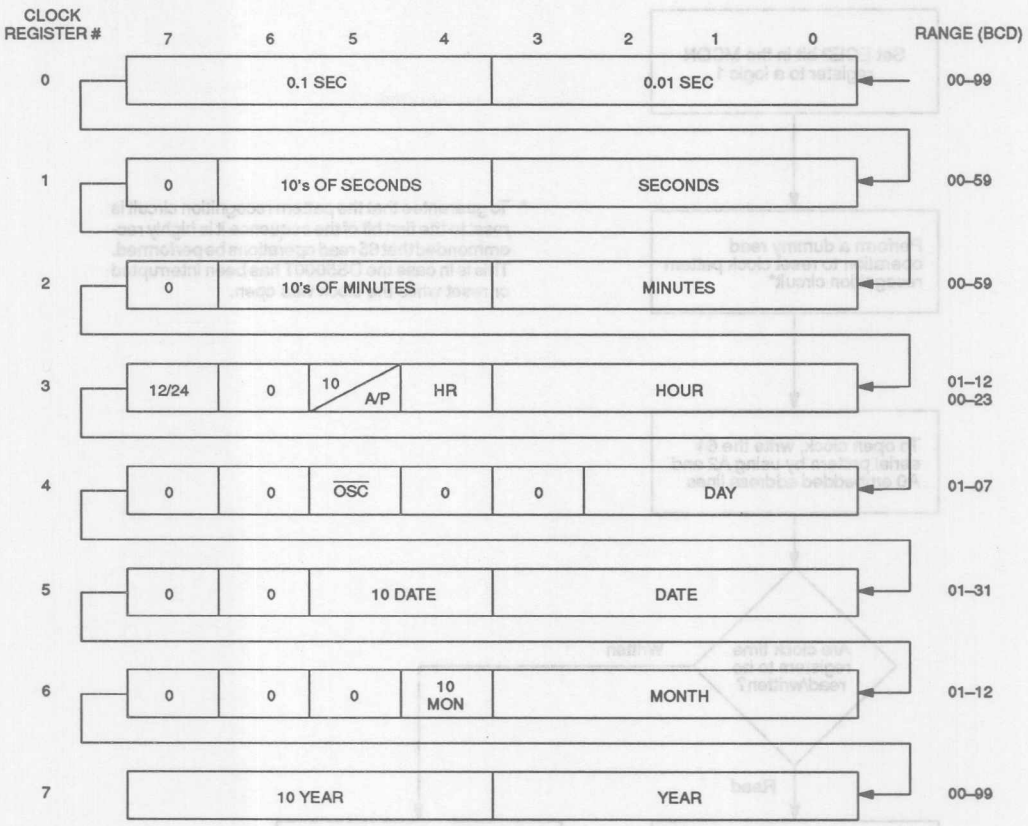


RTC REGISTER ENTRY FLOWCHART Figure 17–3



* To guarantee that the pattern recognition circuit is reset to the first bit of the sequence it is highly recommended that 65 read operations be performed. This is in case the DS5000T has been interrupted or reset while the clock was open.

RTC TIME REGISTERS DESCRIPTION Figure 17–4



REGISTERS

The time information in the RTC is contained in eight registers that are each 8 bits long. After the 64-bit recognition pattern has been received, data in these registers is accessed one bit at a time which is shown conceputally in Figure 17–4. It is recommended that data written to the RTC be handled in groups of 8 bits corresponding to the register bytes in order to prevent erroneous results.

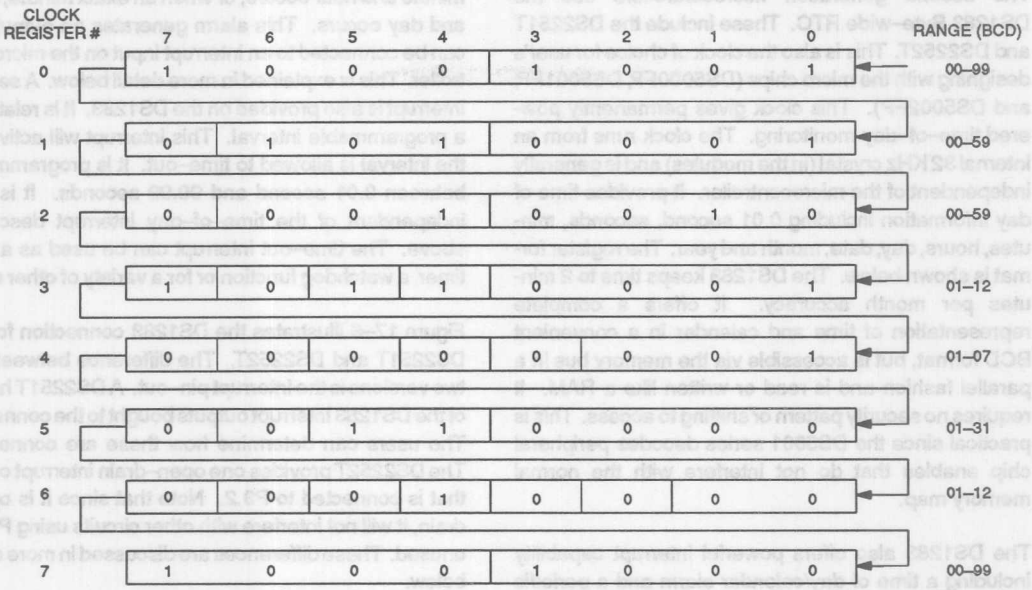
Register data is always in the BCD format except for the hours register (register 3) whose format changes depending upon the state of bit 7. If bit 7 is high, the 12-hour mode is selected and bit 5 of the hours register becomes an AM/PM indicator; if bit 7 is low, the 24-hour mode is selected and bit 5 becomes the second 10-hour bit (20–23 hours). Figure 17–5 contains examples that illustrate the content of these registers for different modes and times.

SPECIAL BITS

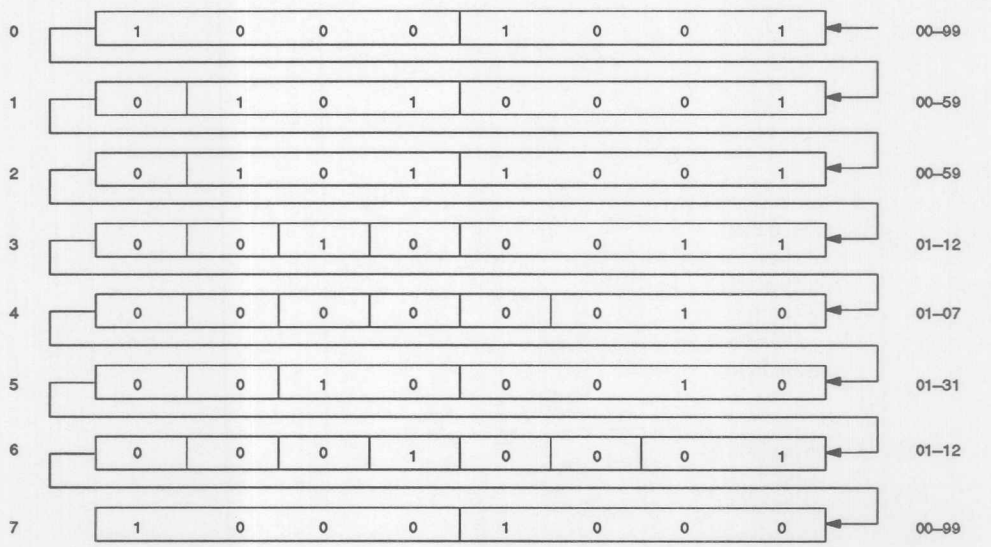
Bit 5 of the days register (register 4) is the control bit for the clock micropower oscillator. Clearing bit 5 to a logic 0 enables the oscillator for normal operation; setting bit 5 to a logic 1 disables the oscillator and halts the timekeeping. It is recommended that bit 5 always be cleared to 0.

Register locations shown as logic 0's in Figure 17–4 will always return a 0 when being read. Write operations to these bit locations are ignored by the clock and have no effect on its operation.

TIME REGISTER EXAMPLES Figure 17–5



The time indicated is 11 o'clock PM, 12 mintues, 51.89 seconds. The date indicated is Sunday, October 30th, 1988.



The time indicated is 2300 hour, 59 minutes, 51.89 seconds. The date indicated is Monday, November 22nd, 1988.

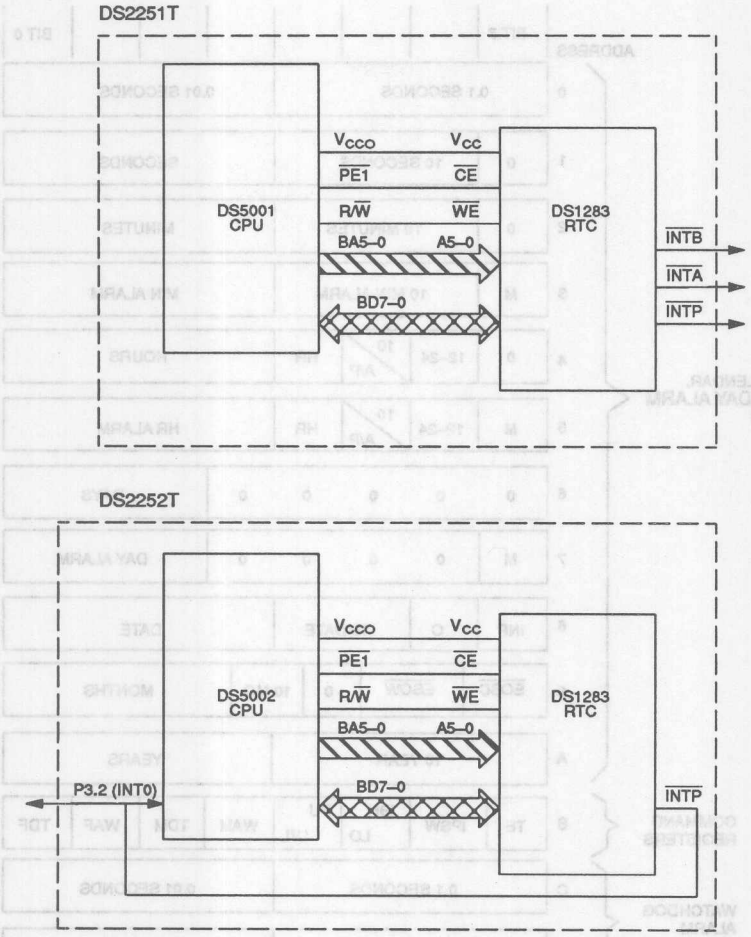
DS1283 BYTE–WIDE CLOCK

The second generation microcontrollers use the DS1283 Byte–wide RTC. These include the DS2251T and DS2252T. This is also the clock of choice for user's designing with the micro chips (DS5000FP, DS5001FP, and DS5002FP). This clock gives permanently powered time–of–day monitoring. The clock runs from an internal 32 KHz crystal (in the modules) and is generally independent of the microcontroller. It provides time of day information including 0.01 second, seconds, minutes, hours, day, date, month and year. The register format is shown below. The DS1283 keeps time to 2 minutes per month accuracy. It offers a complete representation of time and calendar in a convenient BCD format, but is accessible via the memory bus in a parallel fashion and is read or written like a RAM. It requires no security pattern or shifting to access. This is practical since the DS5001 series decodes peripheral chip enables that do not interfere with the normal memory map.

The DS1283 also offers powerful interrupt capability including a time of day/calendar alarm and a periodic interval time–out. The alarm can be set for once per minute, when an exact minute occurs, when an exact minute and hour occurs, or when an exact minute, hour, and day occurs. This alarm generates an output that can be connected to an interrupt input on the microcontroller. This is explained in more detail below. A second interrupt is also provided on the DS1283. It is related to a programmable interval. This interrupt will activate if the interval is allowed to time–out. It is programmable between 0.01 second and 99.99 seconds. It is also independent of the time–of–day interrupt described above. The time–out interrupt can be used as a third timer, a watchdog function or for a variety of other uses.

Figure 17–6 illustrates the DS1283 connection for the DS2251T and DS2252T. The difference between the two versions is the interrupt pin–out. A DS2251T has all of the DS1283 interrupt outputs bought to the connector. The users can determine how these are connected. The DS2252T provides one open–drain interrupt output that is connected to P3.2. Note that since it is open–drain, it will not interfere with other circuits using P3.2 if unused. These differences are discussed in more detail below.

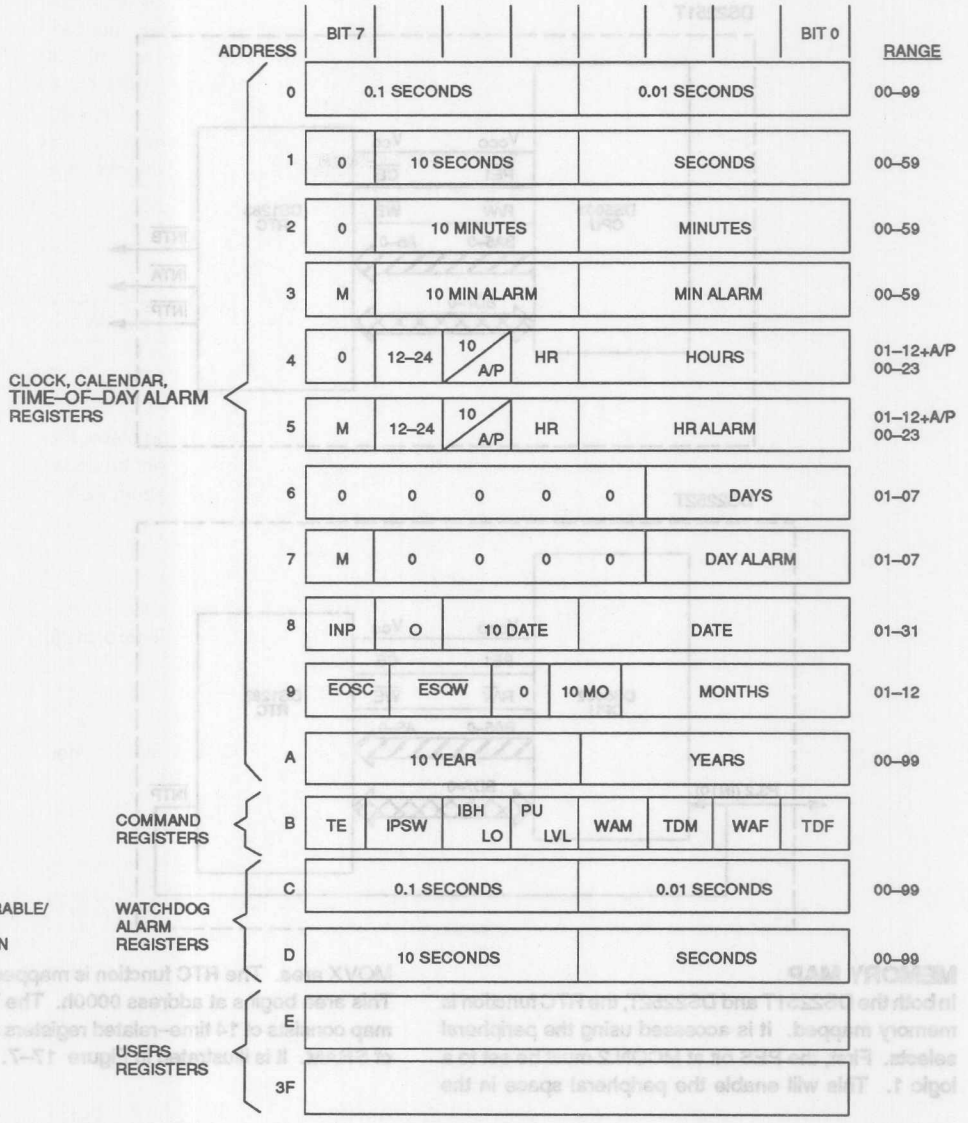
DS2251T/DS2252T RTC BLOCK DIAGRAM Figure 17–6



MEMORY MAP

In both the DS2251T and DS2252T, the RTC function is memory mapped. It is accessed using the peripheral selects. First, the PES bit at MCON.2 must be set to a logic 1. This will enable the peripheral space in the MOVX area. The RTC function is mapped under $\overline{PE1}$. This area begins at address 0000h. The Timekeeping map consists of 14 time–related registers and 50 bytes of SRAM. It is illustrated in Figure 17–7.

REAL–TIME CLOCK MEMORY MAP Figure 17–7



The time, calendar, and alarms are controlled by the information in these 14 registers. In particular, the Command register controls most functions. This is described in Figure 17–8. There are two additional bits that deserve mention. These reside in the register at address 09h. Bit 7 is $\overline{EOSC}$, which enables the Timekeeping oscillator if set to a 0. Lithium lifetime can be preserved by disabling the oscillator when it is not needed and power is not present. Note the user's software should enable the oscillator, as it should be off for shipping. If the oscillator is off, a user can read or write to the Timekeeping register, but the time value will not change. Bit 6 of the same register is the $\overline{ESQW}$ bit. This controls the timekeeper 1024 Hz SQW output. The SQW signal is available on the DS2251T. When it is enabled, it drives a square wave of 1024 Hz. When disabled, it is tri–state so it will not interfere with other uses of a port pin.

REAL–TIME CLOCK COMMAND REGISTER Figure 17–8

RTC COMMAND Register Address 0BH

TE	IPSW	IBH/L0	PU/LVL	WAM	TDM	WAF	TDF

CMD.7: **TE**

Transfer Enable

To avoid updating of time registers while a read is taking place, the update may be frozen. Setting the TE bit to a logic 0 will prevent an update of the user–readable registers from the actual time of day. Setting TE to a logic 1 will enable updates every 0.01 seconds.

CMD.6: **IPSW**

Interrupt Switch

When set to a logic 1, $\overline{INTP}$ will be assigned to time of day alarm and $\overline{INTB}$ will be assigned to the periodic time–out. When set to a logic 0, the functions are reversed.

CMD.5: **IBHL**

INTB H/L

When set to logic 1, the $\overline{INTB}$ will source current (active high). When set to a logic 0, $\overline{INTB}$ will sink current (active low).

CMD.4: **PU/LVL**

Pulse/Level

When set to a logic 1, $\overline{INTP}$ will sink current for approximately 3 ms when it is activated. $\overline{INTB}$ will sink or source (as set by IBLH) for 3 ms. When set to a logic 0, the interrupt pins will signal with a continuous level.

CMD.3: **WAM**

Watchdog Alarm Mask

When set to a logic 1, the watchdog countdown timer interrupt will be disabled. When set to a logic 0, the countdown interrupt is enabled.

CMD.2: **TDM**

Time of Day Alarm Mask

When set to a logic 1, the time of day interrupt is disabled. When set to a logic 0, the time of day alarm is enabled.

CMD.1: **WAF**

Watchdog Alarm Flag

This bit will be set to a logic 1 by the DS1283 when a watchdog time–out occurs. WAF is reset by reading or writing either of the countdown registers.

CMD.0: **TDF**

Time of Day Alarm Flag

This bit is set to a logic 1 by the DS1283 when a time of day alarm occurs. It is cleared by reading or writing any time of day alarm register (register 3, 5, or 7).

RTC INTERRUPTS

The DS1283 provides two interrupt functions. They are time–of–day alarm and a watchdog alarm. The watchdog alarm is a user programmed periodic interval time–out. It is programmed using registers 0Ch and 0Dh. The time–of–day alarm is controlled by the registers at locations 03h, 05h, and 07h as well as the command register. The alarm registers relate to similar time registers. The alarm works by matching the time to the selected alarm according to the mask bits. These are the MSBs of the respective alarm registers. The mask determines if that register is used in the alarm match or is a don't care. There are four valid selections shown in Figure 17–9.

ALARM MASKBIT OPERATION Figure 17–9

MASK			ALARM CONDITION
Minutes	Hours	Days	
1	1	1	Alarm once per minute.
0	1	1	Alarm when minutes match time.
0	0	1	Alarm when minutes and hours match time.
0	0	0	Alarm when minutes, hours, and days match time.

Note: Other mask bit combinations produce illogical operation and should be avoided.

The DS1283 provides three interrupt outputs called $\overline{INTA}$, $\overline{INTB}$, and $\overline{INTP}$. $\overline{INTP}$ is an open–drain representation of $\overline{INTA}$ that can also be forced active. It has no other operational function. Either $\overline{INTA}$ and $\overline{INTB}$ can be assigned to either interrupt function. That is, $\overline{INTA}$ can be the time–of–day alarm or the time–out interval alarm. When $\overline{INTA}$ serves as one function, $\overline{INTB}$ is automatically the other. $\overline{INTP}$ always tracks with $\overline{INTA}$. This allows the RTC interrupts to use only one interrupt pin on the micro if the interrupts will not be used simultaneously. In the DS2251T, all three interrupt pins are available at the connector. The user connects these to the micro port pins of choice. In the DS2252T, only one interrupt signal is available. It is $\overline{INTP}$ and is connected to P3.2 ($\overline{INT0}$). Since $\overline{INTP}$ is an open–drain signal, it will not interfere if not used. When activated, $\overline{INTP}$ will pull P3.2 low. As described above, the interrupt functions can be switched so either is issued via $\overline{INTP}$ ($\overline{INTA}$)

APPLICATION: USING THE RTC ON A DS5000T

The DS5000T or DS2250T uses the DS1215 Phantom type real–time clock (RTC). This clock is basically a serial device that uses a single address bit as an input and a single data bus bit as an output. The following program is an example of how to use this clock. It provides a serial port interface allowing a user to set and read the time of day. Note that the serial port setup expects 9600 baud communication and an 11.0592 MHz crystal. If a user's application uses different values, this setup must be modified. All of the timekeeping subroutines can be incorporated into a user's program by removing the command interface and serial port setup.

Programmer's note: In the Write subroutine at the end of this example program, there is one unusual statement. The action of writing a byte to the RTC is actually done using a read instruction (MOVX A, @DPTR). This is because a write instruction would write to the RAM under $\overline{CE2}$ if one were present. Since the DS1215 is configured to use A2 as a write enable and A0 as the data bit, this instruction is acceptable.

```
; Program DEMODS5T
;
; This program responds to commands received over the serial
; I/O port to send or receive the date/time information between
; the DS1215 in the DS5000T and the serial I/O port.  This allows
; an external program or user to access the date/time information.
;
; The program first sets up the serial port for transmission at
; 9600 baud with eight data bits, no parity, and one stop bit.
;
; Next, the program begins execution of a loop waiting for an
; instruction from the serial port.  Two valid instructions, R and W,
; are recognized.
;
; Receipt of the R character causes the DEMODS5T program
; to read eight bytes of date/time information from the DS1215
; and send them out over the serial port.
;
; Receipt of the W character causes the DEMODS5T program
; to wait for eight bytes of date/time information from the serial
; port and write them to the DS1215.
;
; Any other byte received from the serial port is incremented and
; then sent back out to the serial port.
;
;
PCON            equ     87H
MCON            equ     0C6H
TA              equ     0C7H
;
                cseg    at  0
                sjmp    START
                cseg    at  30H
START:                          ;Initialization.
  mov           TA,     #0AAH     ;Timed
  mov           TA,     #55H      ;access.
  mov           PCON,   #0        ;Reset watchdog timer.
  mov           MCON,   #0F8H     ;Turn off CE2 for memory access.
```

```
  lcall     CLOSE          ;Close date/time registers.

  mov       IE,    #0
  mov       TMOD,  #20H    ;Initialize the
  mov       TH1,   #0FAH   ;serial port
  mov       TL1,   #0FAH   ;for 9600
  orl       PCON,  #80H    ;baud.
  mov       SCON,  #52H
  mov       TCON,  #40H
L:
  jnb       RI,    L       ;Wait for character.
  clr       RI             ;Clear the receiver.
  mov       A,     SBUF    ;Load in the character.
  cjne      A, #'R;, H     ;Skip if not a read.
  lcall     OPEN           ;Set up to read date/time.
  mov       B      #8      ;Set up to send 8 bytes.
F:
  lcall     RBYTE          ;Read a byte of date/time.
G:
  jnb       TI,    G       ;Wait for end of previous send.
  clr       TI             ;Clear transmitter.
  mov       SBUF,  A       ;Send out the byte.
  djnz      B,     F       ;Loop for 8 bytes.
  sjmp      L              ;Return to main loop.
H:
  cjne      A, #'W', J     ;Skip if not a write.
  lcall     OPEN           ;Set up to read date/time.
  mov       B,     #8      ; Set up to receive 8 bytes.
I:
  jnb       RI,    I       ;Wait to receive a byte.
  clr       RI             ;Clear the receiver.
  mov       A,     SBUF    ;Bring in the byte.
  lcall     WBYTE          ;Write a byte of date/time.
  djnz      B,     I       ;Loop for 8 bytes.
  sjmp      L              ;Return to main loop.
J:
  jnb       TI,    J       ;If it is neither an R nor a W
  clr       TI             ;increment the character,
  inc       A              ;send it back out to the
  mov       SBUF,  A       ;serial port, and then
  sjmp      L              ;return to the main loop.
;
;
; SUBROUTINE TO OPEN THE CLOCK/CALENDAR (ECC)
;
; This subroutine executes the sequence of reads and writes which
; is required in order to open communication with the timekeeper.
; The subroutine returns with the timekeeper opened for data
; access with both the accumulator and B register modified.
;
```

```
OPEN:           LCALL       CLOSE           ;Make sure it is closed.
                MOV         B,#4            ;Set pattern period count.
                MOV         A,#0C5H         ;Load first byte of pattern.
OPENA:          LCALL       WBYTE           ;Send out the byte.
                XRL         A,#0FFH         ;Generate next pattern byte.
                LCALL       WBYTE           ;Send out the byte.
                SWAP        A               ;Generate next pattern byte.
                DJNZ        B,OPENA         ;Repeat until 8 bytes sent.
                RET                         ;Return.
;
;*********************************
;*** SUBROUTINE TO CLOSE THE RTC
;*********************************
;
; This subroutine insures that the registers of the timekeeper
; are closed by executing 9 successive reads of the date and time
; registers.  The subroutine returns with both the accumulator
; and the B register modified.
;
CLOSE:          MOV         B,#9            ;Set up to read 9 bytes.
CLOSEA:         LCALL       RBYTE           ;Read a byte.
                DJNZ        B,CLOSEA        ;Loop for 9 byte reads.
                RET                         ;Return.
;
;***********************************
;*** SUBROUTINE TO READ A DAYA BYTE
;***********************************
;
; This subroutine performs a "context switch: to the CE2 data
; space and then reads one byte from the timekeeping device.
; Then it switches back to the CE1 data space and returns
; the byte read in the accumulator, with all other registers
; unchanged.
;
RBYTE:          PUSH        DPL             ;Save the data
                PUSH        DPH             ; pointer on stack.
                PUSH        MCON            ;Save MCON register.
                ORL         MCON,#4         ;Switch to CE2.
                PUSH        B               ;Save the B register.
                MOV         DPL,#4          ;Set up for data input.
                MOV         DPH,#0          ;Set high address byte.
                MOV         B,#8            ;Set the bit count.
LI:             PUSH        AC              ;Save the accumulator.
                MOVX        A,@DPTR         ;Input the data bit.
                RLC         A               ;Move it to carry.
                POP         ACC             ;Get the accumulator.
                RRC         A               ;Save the data bit.
                DJNZ        B,LI            ;Loop for a whole byte.
                POP         B               ;Restore the B register.
                POP         MCON            ;Restore the MCON register.
                POP         DPH             ;Restore the data
```

```
                POP     DPL             ; pointer from stack.
                RET                     ;Return.
;
;************************************
;*** SUBROUTINE TO WRITE A DATA BYTE
;************************************
;
; This subroutine performs a "context switch" to the CE2 data
; space and then writes one byte from the accumulator to the
; timekeeping device.  Then it switches back to the CE1 data
; space and returns with all registers unchanged.
;
WBYTE:          PUSH    DPL             ;Save the data
                PUSH    DPH             ; pointer on stack.
                PUSH    MCON            ;Save the MCON register.
                ORL     MCON,#4         ;Switch to CE2.
                PUSH    B               ;Save the B register.
                MOV     DPH,#0          ;Set high address byte.
                        MOV     B,  #8  ;Set the bit Count.
LO:             PUSH    ACC             ;Save the accumulator.
                ANL     A, #1           ;Set up bit for output.
                MOV     DPL,A           ;Set address to write bit.
                MOVX    A, @DPTR        ;Output the data bit.
                POP     ACC             ;Restore the accumulator.
                RR      A               ;Position next bit.
                DJNZ    B, LO           ;Loop for a whole byte.
                POP     B               ;Restore the B register.
                POP     MCON            ;Restore the MCON register.
                POP     DPH             ;Restore the data
                POP     DPL             ; pointer from stack.
                RET                     ;Return.
;
;****************
; END OF PROGRAM
;****************
;
                END                     ;End of program.
```

APPLICATION: USING THE RTC ON A DS2251T

The DS2251T or DS2252T use the DS1283 Byte–wide type real–time clock (RTC). This clock is accessed in a parallel fashion like a RAM. The user simply writes to the registers to set the time and control functions. The following program is an example of how to use this clock. It provides a serial port interface allowing an user to set and read the time of day. Note that the serial port setup expects 9600 baud communication and an 11.0592 Mhz crystal. If a user's application uses different values, this setup must be modified. All of the timekeeping access is performed in the code under Set Time and Tell Time. The remainder of this program concerns getting data in and out of the serial port for display purposes and has nothing to do with timekeeper access.

```
;          Program DS1283
;
;          This program responds to commands received over the serial
;          port to set the date and time information in the DS1283
;
;          The program first initializes the serial port for communication
;          at 9600 baud with eight data bits, no parity, and one stop bit.
;
;          After setting the date and time, the program begins execution
;          of an infinite loop which sends back the date and time each
;          time a character is received.
;
CR         EQU       0DH
LF         EQU       0AH
;
MCON       EQU       0C6H
TA         EQU       0C7H
           CSEG      AT        RESET
;
           MOV       TA,       #0AAH     ; Timed
           MOV       TA,       #55H      ;    access.
           MOV       PCON,     #0        ; Reset watchdog timer.
           ANL       MCON,     #0FBH     ; Turn off PES for memory access.
           MOV       P2,       #0        ; Clear high byte of address.
                                         ;  for clock access
;
           MOV       IE,       #0
           MOV       TMOD,     #20H      ; Initialize the
           MOV       TH1,      #0FAH     ;    serial port
           MOV       TL1,      #0FAH     ;       for 9600
           ORL       PCON,     #80H      ;          baud.
           MOV       SCON,     #52H
           MOV       TCON,     #40H

;Messages
           MOV       DPTR,     #TEXT0
           LCALL     TEXT_OUT
           LCALL     CHAR_IN
           LCALL     CHAR_OUT
           PUSH      ACC
           MOV       DPTR,     #TEXT3
           LCALL     TEXT_OUT
           POP       ACC
           ANL       A,        #5FH
           CJNE      A, #'Y',  TELL_TIME
```

```
;Set Time
          CLR       A
          MOV       R0,       #0Bh
          LCALL     WBYTE               ; Freeze the registers.
          MOV       DPTR,     #YEAR
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set the year.
          MOV       DPTR,     #MONTH
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set the month.
          MOV       DPTR,     #DAY
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set the day.
          DEC       R0
          MOV       DPTR,     #DAYW
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set day of the week.
          DEC       R0
          MOV       DPTR,     #HOUR
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set the hour.
          DEC       R0
          MOV       DPTR,     #MINUTE
          LCALL     TEXT_OUT
          LCALL     HEX_IN
          LCALL     WBYTE               ; Set the minute.
          CLR       A
          LCALL     WBYTE               ; Set seconds
          LCALL     WBYTE               ;    to 00.00
          MOV       DPTR,     #TRIGGER
          LCALL     TEXT_OUT
          LCALL     CHAR_IN             ; Wait for trigger.
          MOV       A,        #80H
          MOV       R0,       #11
          LCALL     WBYTE               ; Un-freeze the registers.
;
TELL_TIME:
          MOV       DPTR,     #TEXT4
          LCALL     TEXT_OUT            ; Invite user to read time.
CONTINUE:
;
          LCALL     CHAR_IN             ; Wait for read request.
          CLR       A
          MOV       R0,       #11
          LCALL     WBYTE               ; Freeze the registers.
          MOV       DPTR,     #TEXT1
          LCALL     TEXT_OUT
          MOV       R0,       #9
          LCALL     RBYTE               ; Read the month.
          ANL       A,        #1FH      ; Isolate it.
          LCALL     HEX_OUT             ; Display month.
          MOV       A,        #'/'
          LCALL     CHAR_OUT
```

```
            LCALL   RBYTE               ; Read the day of month.
            ANL     A,      #3FH        ; Isolate it.
            LCALL   HEX_OUT             ; Display day of month.
            MOV     A,      #'/'
            LCALL   CHAR_OUT
            MOV     R0,     #10
            LCALL   RBYTE               ; Read the year.
            LCALL   HEX_OUT             ; Display the year.
            MOV     DPTR,   #TEXT2
            LCALL   TEXT_OUT
            MOV     R0,     #4
            LCALL   RBYTE               ; Read the hour.
            DEC     R0
            LCALL   HEX_OUT             ; Display the hour.
            MOV     A,      #':'
            LCALL   CHAR_OUT
            LCALL   RBYTE               ; Read the minute.
            LCALL   HEX_OUT             ; Display the minute.
            MOV     A,      #':'
            LCALL   CHAR_OUT
            LCALL   RBYTE               ; Read the second.
            LCALL   HEX_OUT             ; Display the second.
            MOV     A,      #'.'
            LCALL   CHAR_OUT
            LCALL   RBYTE               ; Read fraction of second.
            LCALL   HEX_OUT             ; Display fraction of second.
            MOV     DPTR,   #TEXT3
            LCALL   TEXT_OUT
            MOV     A,      #80H
            MOV     R0,     #11
            LCALL   WBYTE               ; Un-freeze the registers.
;
            SJMP    CONTINUE            ; Repeat indefinitely.
;
;Utilities

HEX_IN:
            MOV     B,      #0
HEX_LP:
            LCALL   CHAR_IN
            LCALL   CHAR_OUT
            CJNE    A, #0DH,  NOT_CR
            MOV     A,      B
            RET
NOT_CR:
            ADD     A,      #-30H
            JNC     HEX_LP
            CJNE    A,  #10,  $+3
            JC      HEX_XX
            ADD     A,      #-7
            CJNE    A,  #10,  $+3
            JC      HEX_LP
            CJNE    A,  #16,  $+3
            JNC     HEX_LP
HEX_XX:
            XCH     A,      B
            ANL     A,      #0FH
            SWAP    A
```

```
          ORL       A,        B
          MOV       B,        A
          SJMP      HEX_LP
;
HEX_OUT:
          MOV       B,        #2
OUT_LP:
          SWAP      A
          PUSH      ACC
          ANL       A,        #0FH
          CJNE      A,  #10,  $+3
          JC        HEX_OK
          ADD       A,        #7
HEX_OK:
          ADD       A,        #30H
          LCALL     CHAR_OUT
          POP       ACC
          DJNZ      B,        OUT_LP
          RET
;
TEXT_OUT:
          PUSH      ACC
WT1:
          CLR       A
          MOVC      A,        @A+DPTR
          INC       DPTR
          JZ        WT2
          LCALL     CHAR_OUT
          SJMP      WT1
WT2:
          POP       ACC
          RET
;
CHAR_IN:
          JNB       RI,       CHAR_IN
          MOV       A,        SBUF
          CLR       RI
          RET
;
CHAR_OUT:
          JNB       TI,       CHAR_OUT
          MOV       SBUF,     A
          CLR       TI
          RET
;
RBYTE:
          PUSH      MCON                ; Save MCON register.
          ORL       MCON,     #4        ; Switch to PES.
          MOVX      A,        @R0       ; Read the register.
          DEC       R0                  ; Decrement the pointer.
          POP       MCON                ; Restore MCON register.
          RET                           ; Return.
;
WBYTE:
          PUSH      MCON                ; Save MCON register.
          ORL       MCON,     #4        ; Switch to PES.
          MOVX      @R0,      A         ; Read the register.
          DEC       R0                  ; Decrement the pointer.
```

```
        POP     MCON            ; Restore MCON register.
        RET                     ; Return.
;
YEAR:
        DB      CR,LF,'YEAR (0 - 99)   :  ',0
MONTH:
        DB      CR,LF,'MONTH (1 - 12) :  ',0
DAY:
        DB      CR,LF,'DAY OF MONTH    :  ',0
DAYW:
        DB      CR,LF,'DAY OF WEEK     :  ',0
HOUR:
        DB      CR,LF,'HOUR (0 - 23)   :  ',0
MINUTE:
        DB      CR,LF,'MINUTE (0 - 59):  ',0
TRIGGER:
        DB      CR,LF,'PRESS ANY KEY TO SET THIS TIME',CR,LF,0
TEXT0:
        DB      CR,LF,'****** DALLAS SEMICONDUCTOR *******'
        DB      CR,LF,'DS1283 SAMPLE DEMONSTRATION PROGRAM',CR,LF
        DB      CR,LF,'DO YOU WANT TO SET THE TIME (Y/N) ? ',0
TEXT1:
        DB      CR,LF,'DATE:  ',0
TEXT2:
        DB      CR,LF,'TIME:  ',0
TEXT3:
        DB      CR,LF,0
TEXT4:
        DB      CR,LF,'PRESS ANY KEY TO READ THE DATE AND TIME'
        DB      CR,LF,0
;
        END                     ; End of Program.
```

ductor since 1987. It has proven to be reliable and meets its published specifications. However, in the course of development, several common difficulties can be experienced. These are frequently the result of designing around the Soft Micro as through it were exactly an 8051. To reduce these difficulties, Dallas Semiconductor has gathered the common problems in this troubleshooting guide. These are the result of thousands of application questions and represent the most likely sources of trouble. The following section is organized by symptom. This is useful for the developer that encounters trouble. If the symptom listed below is encountered, try the suggested remedies. If these fail, the Dallas applications department will be happy to assist. The next section lists specific do's and don'ts for designing around the Soft Micro. These are largely based on the default practices of 8051 and other microcontroller users.

THE SOFT MICRO APPEARS TO RESET FOR NO REASON

Several features in the Soft Micro can cause a reset. Since these are not part of a standard 8051, a user may be ignoring them.

1. Watchdog Timer. If the Watchdog Timer is enabled, it will cause a reset every 122,800 machine cycles. At 12 MHz, this is 122.8 ms. The Watchdog may be operating even though it was never deliberately enabled. If the Watchdog is not used, deliberately disable it in software as part of the reset vector. If it is used, the code may be missing an opportunity to strobe the Watchdog leading to an accidental reset.
2. Power Supply Glitches. The Soft Micro monitors V_{CC} for a power failure. When power drops below its V_{CCmin} threshold, the micro will reset. Good decoupling can eliminate resets due to noise. A 10 µF and a 0.1 µF capacitor are reasonable values, but actual selections depend on the system. Note, be especially wary of synchronous resets. That is, if every time an event occurs, the micro resets. The event (i.e. turning on a motor) could be causing a dip in V_{CC}.
3. Static. Most microprocessors will loose control during a large static burst. A possible result is to go through a reset. This is especially true if the Watchdog is used.

WRONG TIME

1. Shift register corruption of a DS1215 type clock. When using a DS5000T or DS2250T and ECE2=1, any MOVX will increment the clock pointer. If the micro receives an interrupt while reading the clock, a MOVX done as part of the ISR will alter the clock pointer. Either disable interrupts while in the clock or clear ECE2 as soon as an interrupt occurs.
2. Time is not changing. In any version of a Time Microcontroller, the timekeeper oscillator must be enabled. If the oscillator is off, the time will remain as it was written.

RAM LOSSES DATA WHEN POWERED DOWN

The lithium cell is drained. Under loading, the lithium cell has insufficient capacity to create a voltage that sustains data in the absence of power. This could occur if a negative voltage (below –0.3V) has been applied to the part on any pin. Look for undershoots on power or signals. Also, the power could have been applied in reverse polarity or a DS5000 DIP could have been plugged in backwards. If this happens to a module, the part will still work, but will not retain memory. Note : lithium batteries have a very long time constant. Putting the micro on the shelf for one to two weeks may restore enough voltage to battery back the memory again. The lifetime of such a part would be reduced.

UNABLE TO INVOKE STOP MODE

Unlike the 8051, the STOP bit in the PCON register is Timed Access protected. Existing 8051 code would not use the Timed Access procedure so the STOP mode would not be successfully invoked.

PROGRAM WILL NOT EXECUTE

This is a general category of complaint. In most cases more information is needed. Prior to calling for applications support, check the status of ALE, $\overline{PSEN}$, XTAL2, Ports, and RST. Also, try adding instructions that write a value to the ports to see which sections of code are being run and which are not (like using print statements). The following is a list of some common reasons that the program will not execute.

$\overline{EA}$ is floating

The $\overline{EA}$ pin has an internal pull down resistor. If $\overline{EA}$ is allowed to float, it assumes an active state which prevents using NVRAM. When $\overline{EA}$=0, the micro will use the Expanded Bus on Ports 0 and 2. Connect $\overline{EA}$ directly to +5V for proper operation.

Crystal is not running

Check the capacitance used with the crystals. Approximately 20–40 pF is typical. Take note of any stray capacitance that could cause the actual loading to rise.

Memory map is not configured

If the developer has not used the Bootstrap Loader to select the correct memory map, it may be incompatible with the software. For example, a DS2251 with 64K of memory could be configured with a 32K range. Code above 32K in NVRAM would not be executed.

No Stack

C programmers frequently use a large memory model. This places the C stack in MOVX RAM area. The typical default address for compilers is to place the stack at 0000h. The Soft Micro in a Partitionable mode will not have MOVX NVRAM at 0000h. The C start up configuration should be altered to put the stack and any other data variables above the Partition. C programs will typically crash if there is no stack.

Code is written for an 8052

The 8052 family has 256 bytes of on chip RAM and a third timer. The Soft Micro is an 8051 derivative and does not have these resources.

Watchdog is running and unsupported

If the Watchdog is enabled but not supported by software, it will reset the micro at intervals of 122.8 ms at 12 MHz. If writing to an LCD display or similar activity, the initialization may take more time that this. The code would appear not to run since the display would never get it message printed. Make certain the Watchdog is either supported or disabled in software.

The CRC bit is set on a DS5001

If the CRC bit is set, the DS5001 CPU will invoke the Bootstrap Loader on each power up and perform a CRC. If the answer is incorrect against a stored value, the processor will remain in the loader mode. If the user has inadvertently set this bit and is not actually using the CRC, it will surely be incorrect and will invoke the loader on each power up. A program will seem to run (the ROM checking the NVRAM) then stop.

THE MICRO DRAWS TOO MUCH CURRENT IN STOP MODE

The Soft Micro draws approximately 80 µA of I_{CC} in STOP mode. However, the $\overline{EA}$ pin has a resistive load of between 40K to 125K ohms. If $\overline{EA}$ is connected to +5V, this pin will draw between 40 µA to 125 µA. This current can be eliminated by grounding the $\overline{EA}$ pin and locking the micro. When locked, internal logic disregards the state of $\overline{EA}$. Since it is no longer connected to +5V, the micro will only draw its very low I_{CC}.

DATA IS LOST OR CORRUPTED

A common cause is that data in a DS2250–64 or a DS5000FP based system is lost between banks. The ECE2 bit was most likely left active when software was supposed to write to $\overline{CE1}$ memory. The opposite is also possible. When using first generation micro, data crossing between $\overline{CE1}$ and $\overline{CE2}$ must be managed carefully.

DS5000TK KIT DOES NOT RESPOND TO KIT5K SOFTWARE

1. V_{CC} and ground must be supplied via the ribbon cable. Also, to run a program, a crystal must be supplied. The on board oscillator is used only for loading.
2. Cable is broken or a standard phone cable has been used. A standard phone cable has the wrong pin out of the DS5000TK.
3. Incorrect COM port has been selected.
4. Micro is not locked into its ZIF socket.

SERIAL COMMUNICATION FAILS ON A DS5000TK

1. The ribbon cable represents a significant stray capacitance to the crystal pins. The crystal may be running at the wrong frequency. This can be checked by observing ALE which should be 1/6 of the crystal. If it is not, adjust the capacitors to get the correct frequency.
2. The A/B switch is in the wrong position. In position A, the micro serial port is routed via the ribbon cable to the target system. In position B, it goes back to the PC COM port.

THE DEMODS5T PROGRAM DOES NOT WORK

Normally due to the serial communication problems mentioned above. For the demo, the crystal must be 11.0592 MHz and must really be at that frequency. Also, the A/B switch must be in position B so the micro communicates with the PC via the serial port while running code.

DO'S AND DON'TS

This section highlights some common mistakes and points out some helpful hints. Not all of these are applicable to every system.

DON'TS

RC Resets

Do not use an RC circuit for a power–on reset. The Soft Micro does this internally. If the traditional RC circuit is used without a diode, it will expose the RST pin to –5V if power falls faster than the RC time constant. This is due to the capacitor retaining its charge.

Lithium backed signals

Do not connect lithium backed chip enables or signals to non–backed devices. This produces a drain on the lithium cell. On the DS5001 and DS5002, $\overline{PE1}$ and $\overline{PE2}$ as a well as $\overline{CE1}$ – 4 are lithium backed. $\overline{PE3}$ and $\overline{PE4}$ are not backed and can be connected to normal circuits. On the DS5000FP, $\overline{CE1}$ and $\overline{CE2}$ as well as BA13 and BA14 are lithium backed.

Negative Voltage

Do not allow negative voltage to contact the micro. This includes under shoots on power or ports, static, plugging in backwards, or applying a signal without a common ground reference.

DO'S

Static Protection

Do use Schottky diodes and resistor on port pins that are available at the outside world. Static can represent a negative voltage that temporarily collapses the lithium backup.

Loader Port

Do provide a method of in–system loading such as an RS232 transceiver, a connector, and a way to invoke the loader. This is especially important to applications that are encased in epoxy.

Use the Watchdog

All microprocessor systems encounter situations that they can not deal with by design. The Watchdog is the first line of defense. If software runs out of control, it can only do so for the duration of one Watchdog time–out.

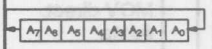
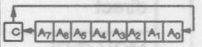
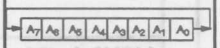
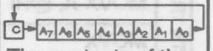
Control power supply

An ideal situation is when the micro controls the power down function. If the power switch actually asks software to turn off the power, then software is never taken by surprise. Also, make certain that the power supply slew rate is longer than 40 uS between V_{CCmin} and +3V.

SECTION 19: INSTRUCTION SET DETAILS

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
ARITHMETIC OPERATION	ADD A, Rn	0	0	1	0	1	n_2	n_1	n_0	28-2F	1	1	(A) = (A) + (Rn)
	ADD A, direct	0	0	1	0	0	1	0	1	25	2	1	(A) = (A) + (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	ADD A, @Ri	0	0	1	0	0	1	1	i	26-27	1	1	(A) = (A) + ((Ri))
	ADD A, #data	0	0	1	0	0	1	0	0	24	2	1	(A) = (A) + #data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	ADDC A, Rn	0	0	1	1	1	n_2	n_1	n_0	38-3F	1	1	(A) = (A)+(C)+(Rn)
	ADDCA,direct	0	0	1	1	0	1	0	1	35	2	1	(A) =
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			(A)+(C)+(direct)
	ADDC A, @Ri	0	0	1	1	0	1	1	i	36-37	1	1	(A) = (A)+(C)+((Ri))
	ADDC A,#data	0	0	1	1	0	1	0	0	34	2	1	(A) = (A)+(C)+#data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	SUBB A, Rn	1	0	0	1	1	n_2	n_1	n_0	98-9F	1	1	(A) = (A)-(C)-(Rn)
	SUBB A,direct	1	0	0	1	0	1	0	1	95	2	1	(A) =
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			(A)-(C)-(direct)
	SUBB A, @Ri	1	0	0	1	0	1	1	i	96-97	1	1	(A) = (A)-(C)-((Ri))
	SUBB A,#data	1	0	0	1	0	1	0	0	94	2	1	(A) = (A)-(C)-#data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	INC A	0	0	0	0	0	1	0	0	04	1	1	(A) = (A) + 1
	INC Rn	0	0	0	0	1	n_2	n_1	n_0	08-0F	1	1	(Rn) = (Rn) + 1
	INC direct	0	0	0	0	0	1	0	1	05	2	1	(direct) = (direct)+1
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	INC @Ri	0	0	0	0	0	1	1	i	06-07	1	1	((Ri)) = ((Ri)) + 1
	INC DPTR	1	0	1	0	0	0	1	1	A3	1	2	(DPTR)=(DPTR)+1
	DEC A	0	0	0	1	0	1	0	0	14	1	1	(A) = (A) - 1
	DEC Rn	0	0	0	1	1	n_2	n_1	n_0	18-1F	1	1	(Rn) = (Rn) - 1
	DEC direct	0	0	0	1	0	1	0	1	15	2	1	(direct) = (direct)-1
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	DEC @Ri	0	0	0	1	0	1	1	i	16-17	1	1	((Ri)) = ((Ri)) - 1
	MUL AB	1	0	1	0	0	1	0	0	A4	1	4	(B_{15-8}), (A_{7-0}) = (A) X (B)
	DIV AB	1	0	0	0	0	1	0	0	84	1	4	(A_{15-8}), (A_{7-0}) = (A) ÷ (B)

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
ARITHMETIC OPER.	DA A	1	1	0	1	0	1	0	0	D4	1	1	Contents of Accumulator are BCD, IF [[(A_{3-0}) > 9] OR [(AC) = 1]] THEN $(A_{3-0}) = (A_{3-0}) + 6$ AND IF [[(A_{7-4}) > 9] OR [(C) = 1]] THEN $(A_{7-4}) = (A_{7-4}) + 6$
LOGICAL OPERATION	ANL A, Rn	0	1	0	1	1	n_2	n1	n_0	58-5F	1	1	(A) = (A) AND (Rn)
	ANL A, direct	0	1	0	1	0	1	0	1	55	2	1	(A) = (A) AND (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	ANL A, @Ri	0	1	0	1	0	1	1	1	56-57	1	1	(A) = (A) AND ((Ri))
	ANL A, #data	0	1	0	1	0	1	0	0	54	2	1	(A)=(A) AND #data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	ANL direct, A	0	1	0	1	0	0	1	0	52	2	1	(direct) = (direct) AND A
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	ANL direct, #data	0	1	0	1	0	0	1	1	53	3	2	(direct) = (direct) AND #data
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 3			
	ORL A, Rn	0	1	0	0	1	n_2	n_1	n_0	48-4F	1	1	(A) = (A) OR (Rn)
	ORL A, direct	0	1	0	0	0	1	0	1	45	2	1	(A) = (A) OR (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	ORL A, @Ri	0	1	0	0	0	1	1	i	46-47	1	1	(A) = (A) OR ((Ri))
	ORL A, #data	0	1	0	0	0	1	0	0	44	2	1	(A) = (A) OR #data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	ORL direct, A	0	1	0	0	0	0	1	0	42	2	1	(direct) = (direct) OR (A)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	ORL direct,#data	0	1	0	0	0	0	1	1	43	3	2	(direct) = (direct) OR #data
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 3			
	XRL A, Rn	0	1	1	0	1	n_2	n_1	n_0	68-6F	1	1	(A) = (A) XOR (Rn)
	XRL A, direct	0	1	1	0	0	1	0	1	65	2	1	(A) = (A) XOR (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	XRL A, @ Ri	0	1	1	0	0	1	1	i	66-67	1	1	(A) = (A) XOR ((Ri))
	XRL A, #data	0	1	1	0	0	1	0	0	64	2	1	(direct) = (direct) XOR #data
		d_7	d6	d5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	XRL direct, A	0	1	1	0	0	0	1	0	62	2	1	(direct) = (direct) XOR (A)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	XRL direct, #data	0	1	1	0	0	0	1	1	63	3	2	(direct) = (direct XOR (A)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 3			
	CLR A	1	1	1	0	0	1	0	0	E4	1	1	(A) = 0
	CPL A	1	1	1	1	0	1	0	0	F4	1	1	(A) = $(\overline{A})$

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
LOGICAL OPERATION	RL A	0	0	1	0	0	0	1	1	23	1	1	 The contents of the accumulator are rotated left by one bit.
	RLC A	0	0	1	1	0	0	1	1	33	1	1	 The contents of the accumulator are rotated left by one bit.
	RR A	0	0	0	0	0	0	1	1	03	1	1	 The contents of the accumulator are rotated right by one bit.
	RRC A	0	0	0	1	0	0	1	1	13	1	1	 The contents of the accumulator are rotated right by one bit.
	SWAP A	1	1	0	0	0	1	0	0	C4	1	1	$(A_{3-0}) = (A_{7-4})$
DATA TRANSFER	MOV A, Rn	1	1	1	0	1	n_2	n_1	n_0	E8-EF	1	1	(A) = (Rn)
	MOV A, direct	1	1	1	0	0	1	0	1	E5	2	1	(A) = (direct)
		a_7	a_6	a_3	a_4	a_3	a_2	a_1	a_0	Byte 2			
	MOV A, @Ri	1	1	1	0	0	1	1	i	E6-E7	1	1	(A) = ((Ri))
	MOV A, #data	0	1	1	1	0	1	0	0	74	2	1	(A) = #data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	MOV Rn, A	1	1	1	1	1	n_2	n_1	n_0	F8-FF	1	1	(Rn) = (A)
	MOV Rn, direct	1	0	1	0	1	n_2	n_1	n_0	A8-AF	2	2	(Rn) = (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	MOV Rn, #data	0	1	1	1	1	n_2	n_1	n_0	78-7F	2	1	(Rn) = #data
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 2			
	MOV direct, A	1	1	1	1	0	1	0	1	F5	2	1	(direct) = (A)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	MOV direct, Rn	1	0	0	0	1	n_2	n_1	n_0	88-8F	2	2	(direct) = (Rn)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	MOV direct1, direct2	1	0	0	0	0	1	0	1	85	3	2	(direct1) = (direct2)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			(source)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 3			(destination)
	MOV direct, @Ri	1	0	0	0	0	1	1	i	86-87	2	2	(direct) = ((Ri))
		a_7	a_6	a_5	a4	a_3	a_2	a_1	a_0	Byte 2			

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
DATA TRANSFER	MOV direct, #data	0	1	1	1	0	1	0	1	75	3	2	(direct) = #data
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 3			
	MOV @Ri, A	1	1	1	1	0	1	1	i	F6-F7	1	1	((Ri)) = A
	MOV @Ri, direct	1	0	1	0	0	1	1	i	A6-A7	2	2	((Ri)) = (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	MOV @Ri, #data	0	1	1	1	0	1	1	i	76-77	2	1	((Ri)) = #data
		d_7	d_6	d_5	d_4	d_3	d_2	d1	d_0	Byte 2			
	MOV DPTR, #data16	1	0	0	1	0	0	0	0	90	3	2	(DPTR) = #data$_{15-0}$
		d_7	d_6	d_5	d4	d_3	d_2	d_1	d_0	Byte 2			(DPH) = #data$_{15-8}$
		d_7	d_6	d_5	d_4	d_3	d_2	d_1	d_0	Byte 3			(DPL) = #data$_{7-0}$
	MOVC A, @A + DPTR	1	0	0	1	0	0	1	1	93	1	2	(A)=((A) + (DPTR))
	MOVC A, @A + PC	1	0	0	0	0	0	1	1	83	1	2	(A) = ((A) + (PC))
	MOVX A, @Ri	1	1	1	0	0	0	1	i	E2-E3	1	2	(A) = ((Ri))
	MOVX @DPTR	1	1	1	0	0	0	0	0	E0	1	2	(A) = ((DPTR))
	MOVX @Ri, A	1	1	1	1	0	0	1	i	F2-F3	1	2	((Ri)) = (A)
	MOVX @DPTR,A	1	1	1	1	0	0	0	0	F0	1	2	((DPTR)) = (A)
	PUSH direct	1	1	0	0	0	0	0	0	C0	2	2	(SP) = (SP) + 1
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			((SP)) = (direct)
	POP direct	1	1	0	1	0	0	0	0	D0	2	2	(direct) = ((SP))
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			(SP) = (SP) - 1
	XCH A, Rn	1	1	0	0	1	n_2	n_1	n_0	C8-CF	1	1	(A) = (Rn)
	XCH A, direct	1	1	0	0	0	1	0	1	C5	2	1	(A) = (direct)
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	XCH A, @Ri	1	1	0	0	0	1	1	i	C6-C7	1	1	(A) = ((Ri))
	XCHD A, @Ri	1	1	0	1	0	1	1	i	D6-D7	1	1	(A_{3-0}) = ((Ri_{3-0}))
BOOLEAN VARIABLE MANIPULATION	CLR C	1	1	0	0	0	0	1	1	C3	1	1	(C) = 0
	CLR bit	1	1	0	0	0	0	1	0	C2	2	1	(bit) = 0
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	SETB C	1	1	0	1	0	0	1	1	D3	1	1	(C) = 1
	SETB bit	1	1	0	1	0	0	1	0	D2	2	1	(bit) = 1
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	CPL C	1	0	1	1	0	0	1	1	B3	1	1	(C) = $(\overline{C})$
	CPL bit	1	0	1	1	0	0	1	0	B2	2	1	(bit) = $(\overline{bit})$
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	ANL C, bit	1	0	0	0	0	0	1	0	82	2	2	(C) = (C) AND (bit)
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
BOOLEAN VARIABLE MANIPULATION	ANL C, $\overline{bit}$	1	0	1	1	0	0	0	0	B0	2	2	(C) = (C) AND ($\overline{bit}$)
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	ORL C, bit	0	1	1	1	0	0	1	0	72	2	2	(C) = (C) OR (bit)
		b_7	b6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	ORL C, $\overline{bit}$	1	0	1	0	0	0	0	0	A0	2	2	(C) = (C) OR ($\overline{bit}$)
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	MOV C, bit	1	0	1	0	0	0	1	0	A2	2	1	(C) = (bit)
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
	MOV bit, C	1	0	0	1	0	0	1	0	92	2	2	(bit) = (C)
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
PROGRAM BRANCHING	ACALL addr 11	a_{10}	a_9	a_8	1	0	0	0	1	Byte 1	2	2	(PC) = (PC) + 2 (SP) = (SP) + 1 ((SP)) = (PC_{7-0}) (SP) = (SP) + 1 ((SP)) = (PC_{15-8}) (PC)=page address
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	LCALL addr 16	0	0	0	1	0	0	1	0	12	3	2	(PC) = (PC) + 3 (SP) = (SP) + 1 ((SP)) = (PC_{7-0}) (SP) = (SP) + 1 ((SP)) = (PC_{15-8}) (PC) = $addr_{15-0}$
		a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 2			
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 3			
	RET	0	0	1	0	0	0	1	0	22	1	2	(PC_{15-8}) = ((SP)) (SP) = (SP) - 1 (PC_{7-0}) = ((SP)) (SP) = (SP) - 1
	RETI	0	0	1	1	0	0	1	0	32	1	2	(PC_{15-8}) = ((SP)) (SP) = (SP) - 1 (PC_{7-0}) = ((SP)) (SP) = (SP) - 1
	AJMP addr 11	a_{10}	a_9	a_8	0	0	0	0	1	Byte 1	2	2	(PC) = (PC) + 2 (PC_{10-0}) = page addr
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 2			
	LJMP addr 16	0	0	0	0	0	0	1	0	02	3	2	(PC) = addr15-0
		a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	Byte 2			
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	Byte 3			
	SJMP rel	1	0	0	0	0	0	0	0	80	2	2	(PC) = (PC) + 2 (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 2			
	JMP @A + DPTR	0	1	1	1	0	0	1	1	73	1	2	(PC) = (A) + (DPTR)
	JZ rel	0	1	1	0	0	0	0	0	60	2	2	(PC) = (PC) + 2 IF (A) = 0 THEN (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 2			
	JNZ rel	0	1	1	1	0	0	0	0	70	2	2	(PC) = (PC) + 2 IF (A) $\neq$ 0 THEN (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 2			

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
PROGRAM BRANCHING	JC rel	0	1	0	0	0	0	0	0	40	2	2	(PC) = (PC) + 2 IF (C) = 1 THEN (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 2			
	JNC rel	0	1	0	1	0	0	0	0	50	2	2	(PC) = (PC) + 2 IF (C) $\neq$ 0 THEN (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 2			
	JB bit, rel	0	0	1	0	0	0	0	0	20	3	2	(PC) = (PC) + 3 IF (bit) = 1 THEN (PC) = (PC) + rel
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	JNB bit, rel	0	0	1	1	0	0	0	0	30	3	2	(PC) = (PC) + 3 IF (bit) = 0 THEN (PC) = (PC) + rel
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	JBC bit, direct rel	0	0	0	1	0	0	0	0	10	3	2	(PC) = (PC) + 3 IF (bit) = 1 THEN (bit) = 0 (PC) = (PC) + rel
		b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	CJNE A, direct rel	1	0	1	1	0	1	0	1	B5	3	2	(PC) = (PC) + 3 IF (direct) < (A) THEN (PC) = (PC) + rel and (C) = 0 OR IF (direct) > (A) THEN (PC) = (PC) + rel and (C) = 1
		a7	a6	a5	ad	a3	a2	a1	a0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	CJNE A, #data rel	1	0	1	1	0	1	0	0	B4	3	2	(PC) = (PC) + 3 IF #data < (A) THEN (PC) = (PC) + rel and (C) = 0 OR IF #data > (A) THEN (PC) = (PC) + rel and (C) = 1
		d7	d6	d5	d4	d3	d2	d1	d0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	CJNE Rn, #data rel	1	0	1	1	1	n_2	n_1	n_0	B8-BF	3	2	(PC) = (PC) + 3 IF #data < (Rn) THEN (PC) = (PC) + rel and (C) = 0 OR IF #data > (Rn) THEN (PC) = (PC) + rel and (C) = 1
		d7	d6	d5	d4	d3	d2	d1	d0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			
	CJNE @Ri, #data rel	1	0	1	1	0	1	1	i	B6-B7	3	2	(PC) = (PC) + 3 IF #data < ((Ri)) THEN (PC) = (PC) + rel and (C) = 0 OR IF #data > ((Ri)) THEN (PC) = (PC) + rel and (C) = 1
		d7	d6	d5	d4	d3	d2	d1	d0	Byte 2			
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0	Byte 3			

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
PROGRAM BRANCHING	DJNZ Rn, rel	1 r_7	1 r_6	0 r_5	1 r_4	1 r_3	n_2 r_2	n_1 r_1	n_0 r_0	D8-Df Byte 2	2	2	(PC) = (PC) + 2 (Rn) = (Rn) - 1 IF (Rn) 0 THEN (PC) = (PC) + rel
	DJNZ direct rel	1 a_7 r_7	1 a_6 r_6	0 a_5 r_5	1 a_4 r_4	0 a_3 r_3	1 a_2 r_2	0 a_1 r_1	1 a_0 r_0	D5 Byte 2 Byte 3	3	2	(PC) = (PC) + 3 (direct) = (direct) - 1 IF (direct)≠0 THEN (PC) = (PC) + rel
	NOP	0	0	0	0	0	0	0	0	00	1	1	(PC) = (PC) + 1

	MNEMONIC	INSTRUCTION CODE								HEX	BYTE	CYCLE	EXPLANATION
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0				
PROGRAM INSTRUCTIONS	DJNZ Rn, rel	1	1	0	1	1	r_2	r_1	r_0	D8-DF Byte 2	2	2	(PC) = (PC) + 2 (Rn) = (Rn) - 1 IF (Rn) 0 THEN (PC) = (PC) + rel
		r_7	r_6	r_5	r_4	r_3							
	DJNZ direct rel	1	1	0	1	0	1	0	1	D5 Byte 2 Byte 3	3	2	(PC) = (PC) + 3 (direct) = (direct) - 1 IF (direct)≠0 THEN (PC) = (PC) + rel
		a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0				
		r_7	r_6	r_5	r_4	r_3	r_2	r_1	r_0				
	NOP	0	0	0	0	0	0	0	0	00	1	1	(PC) = (PC) + 1

SOFT MICROCONTROLLER FAMILY DATA SHEETS

SOFT MICROCONTROLLER
FAMILY DATA SHEETS

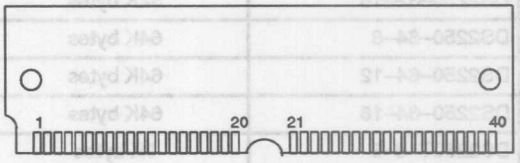
DALLAS SEMICONDUCTOR

DS2250(T)
Soft Microcontroller

FEATURES

- 8–bit 8051 compatible uC adapts to task–at–hand:
  - 8, 32, or 64 Kbytes of nonvolatile RAM for program and/or data memory storage
  - Initial downloading of software in end system via on–chip serial port
  - Capable of modifying its own program and/or data memory in end use
- Crashproof operation:
  - Maintains all nonvolatile resources for 10 years in the absence of V_{CC}
  - Power–fail reset
  - Early warning power–fail interrupt
  - Watchdog timer
- Software Security Feature:
  - Executes encrypted software to prevent unauthorized disclosure
- On–chip, full–duplex serial I/O ports
- Two on–chip timer/event counters
- 32 parallel I/O lines
- Compatible with industry standard 8051 instruction set
- Optional Permanently Powered Real–Time Clock (DS2250T)

PIN ASSIGNMENT



40-Pin SIMM

DESCRIPTION

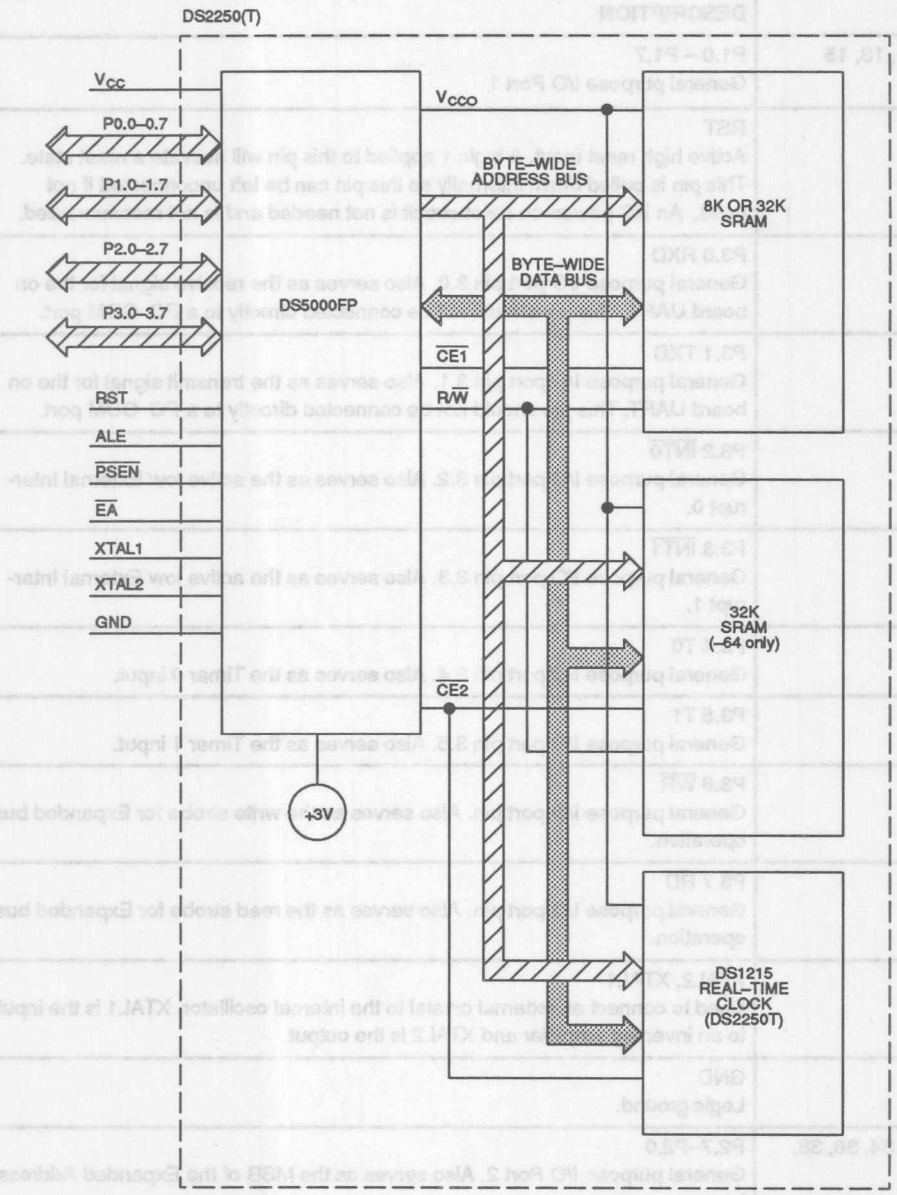
The DS2250(T) Soft Microcontroller is a fully 8051 compatible 8–bit CMOS microcontroller that offers "softness" in all aspects of its application. This is accomplished through the comprehensive use of nonvolatile technology to preserve all information in the absence of system V_{CC}. The internal program/data memory space is implemented using 8, 32, or 64 Kbytes of nonvolatile CMOS SRAM. Furthermore, internal data registers and key configuration registers are also nonvolatile. An optional real–time clock gives permanently powered timekeeping. The clock keeps time to a hundredth of a second using an on–board crystal. All nonvolatile memory and resources are maintained for over 10 years at room temperature in the absence of power.

ORDERING INFORMATION

PART NUMBER	RAM SIZE	MAX CRYSTAL SPEED	TIMEKEEPING?
DS2250–8–8	8K bytes	8 MHz	No
DS2250–8–12	8K bytes	12 MHz	No
DS2250–8–16	8K bytes	16 MHz	No
DS2250–32–8	32K bytes	8 MHz	No
DS2250–32–12	32K bytes	12 MHz	No
DS2250–32–16	32K bytes	16 MHz	No
DS2250–64–8	64K bytes	8 MHz	No
DS2250–64–12	64K bytes	12 MHz	No
DS2250–64–16	64K bytes	16 MHz	No
DS2250T–8–8	8K bytes	8 MHz	Yes
DS2250T–8–12	8K bytes	12 MHz	Yes
DS2250T–8–16	8K bytes	16 MHz	Yes
DS2250T–32–8	32K bytes	8 MHz	Yes
DS2250T–32–12	32K bytes	12 MHz	Yes
DS2250T–32–16	32K bytes	16 MHz	Yes
DS2250T–64–8	64K bytes	8 MHz	Yes
DS2250T–64–12	64K bytes	12 MHz	Yes
DS2250T–64–16	64K bytes	16 MHz	Yes

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pinout, and electrical specifications.

DS2250(T) BLOCK DIAGRAM Figure 1



PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
1, 3, 5, 7, 9, 11, 13, 15	P1.0 – P1.7 General purpose I/O Port 1
17	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally so this pin can be left unconnected if not used. An RC power–on reset circuit is not needed and is not recommended.
19	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should not be connected directly to a PC COM port.
21	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should not be connected directly to a PC COM port.
23	P3.2 $\overline{\text{INT0}}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
25	P3.3 $\overline{\text{INT1}}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
27	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
29	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
31	P3.6 $\overline{\text{WR}}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
33	P3.7 $\overline{\text{RD}}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
35, 37	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
39	GND Logic ground.
26, 28, 30, 32, 34, 36, 38, 40	P2.7–P2.0 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.

PIN NUMBER	DESCRIPTION
24	$\overline{PSEN}$ Program Store Enable. This active low signal is used to enable an external program memory when using the Expanded bus. It is normally an output and should be unconnected if not used. $\overline{PSEN}$ also is used to invoke the Bootstrap Loader. At this time, $\overline{PSEN}$ will be pulled down externally. This should only be done once the DS2250 is already in a reset state. The device that pulls down should be open drain since it must not interfere with $\overline{PSEN}$ under normal operation.
22	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch. When using a parallel programmer, this pin also assumes the $\overline{PROG}$ function for programming pulses.
20	$\overline{EA}$ External Access. This pin forces the DS2250 to behave like an 8031. No internal memory (or clock) will be available when this pin is at a logic low. Since this pin is pulled down internally, it should be connected to +5V to use NVRAM. In an parallel programmer, this pin also serves as V_{PP} for super voltage pulses.
4, 6, 8, 10, 12, 14, 16, 18	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull–ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
2	V_{CC} +5 volts.

INSTRUCTION SET

The DS2250 executes an instruction set which is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages which have been written for the 8051 are compatible with the DS2250, including cross–assemblers, high–level language compilers, and debugging tools. Note that the DS2250 is functionally identical to the DS5000 except for package and the 64K memory option.

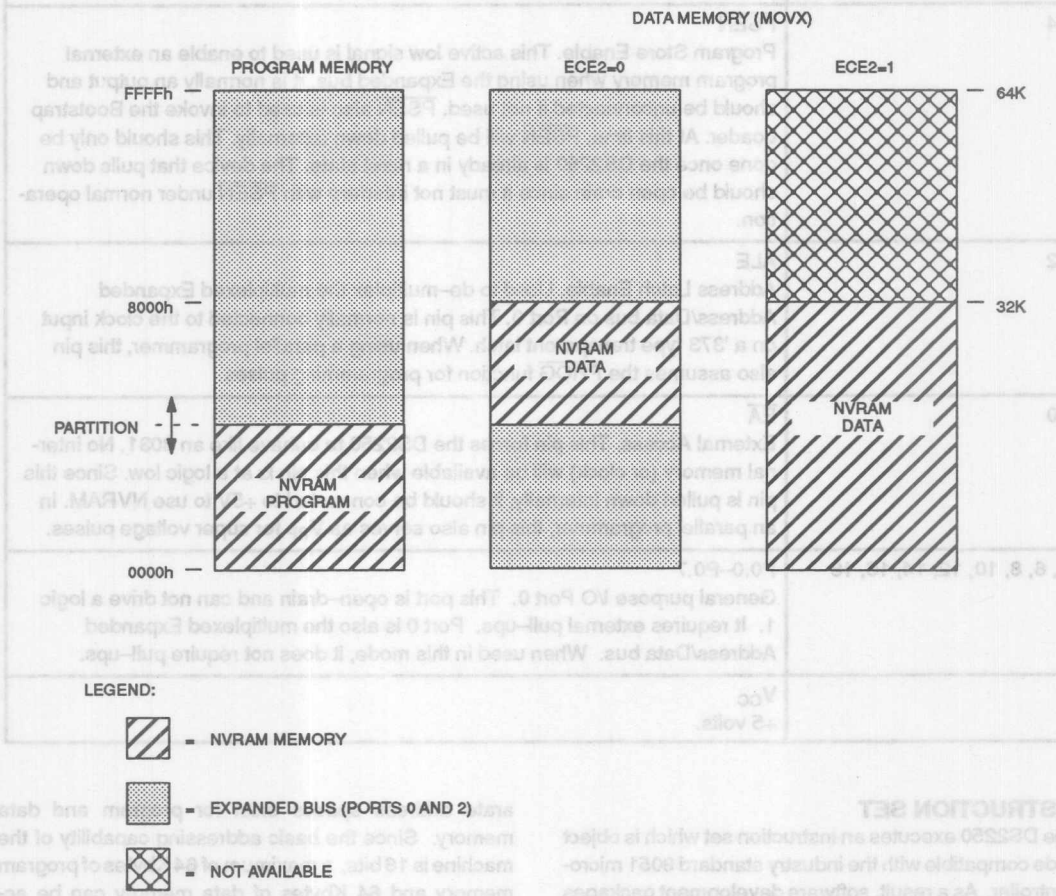
A complete description for the DS2250 instruction set is available in the User's Guide section of the Soft Microcontroller Data Book.

MEMORY ORGANIZATION

Figure 2 illustrates the address spaces which are accessed by the DS2250. As illustrated in the figure, separate address spaces exist for program and data memory. Since the basic addressing capability of the machine is 16 bits, a maximum of 64 Kbytes of program memory and 64 Kbytes of data memory can be accessed by the DS2250 CPU. The 8K or 32K byte RAM area inside of the DS2250 can be used to contain both program and data memory. A second 32K RAM is available for data only.

The Real–time Clock (RTC) in the DS2250T is reached in the memory map by setting a SFR bit. The MCON.2 bit (ECE2) is used to select an alternate data memory map. While ECE2=1, all MOVXs will be routed to this alternate memory map. The real–time clock is a serial device that resides in this area. A full description of the RTC access and example software is given in the User's Guide section of the Soft Microcontroller Data Book.

DS2250 MEMORY MAP Figure 2



PROGRAM LOADING

The Program Load Modes allow initialization of the NVRAM Program/Data Memory. This initialization may be performed in one of two ways:

1. Serial Program Loading which is capable of performing Bootstrap Loading of the DS2250(T). This feature allows the loading of the application program to be delayed until the DS2250(T) is installed in the end system.
2. Parallel Program Load cycles which perform the initial loading from parallel address/data information presented on the I/O port pins. this mode is timing–set compatible with the 87C51H microcontroller programming mode.

The DS2250 is placed in its Program Load configuration by simultaneously applying a logic 1 to the RST pin and forcing the $\overline{\text{PSEN}}$ line to a logic 0 level. Immediately following this action, the DS2250 will look for a parallel Program Load pulse, or a serial ASCII carriage return (0DH) character received at 9600, 2400, 1200, or 300 bps over the serial port.

The hardware configurations used to select these modes of operation are illustrated in Figure 3.

PROGRAM LOADING CONFIGURATIONS Figure 3

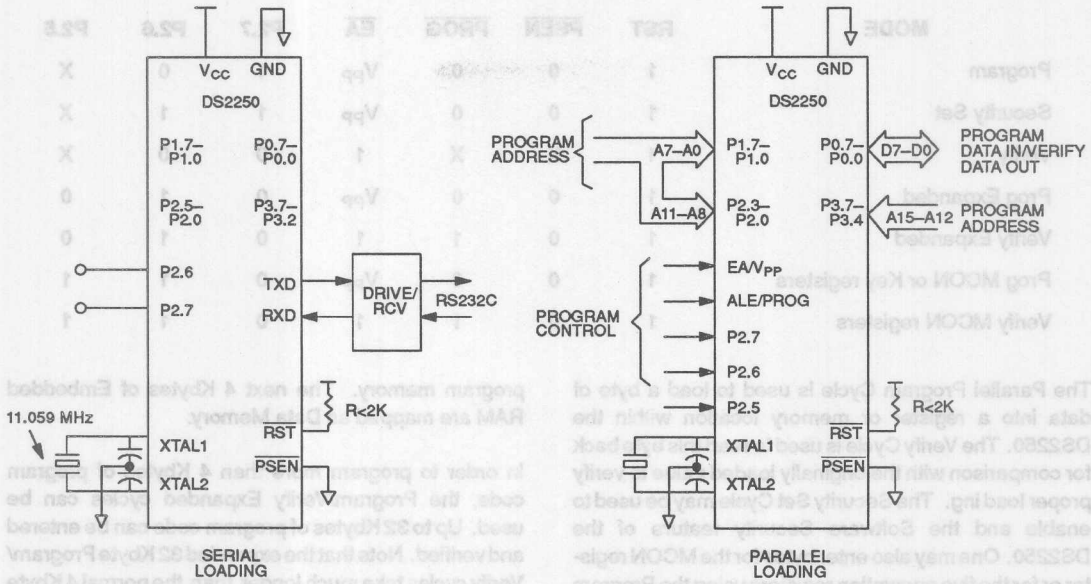


Table 1 summarizes the selection of the available Parallel Program Load cycles. The timing associated with these cycles is illustrated in the electrical specs.

SERIAL BOOTSTRAP LOADER

The Serial Program Load Mode is the easiest, fastest, most reliable, and most complete method of initially loading application software into the DS2250 nonvolatile RAM. Communication can be performed over a standard asynchronous serial communications port. A typical application would use a simple RS232C serial interface to program the DS2250 as a final production procedure. The hardware configuration which is required for the Serial Program Load mode is illustrated in Figure 3. Port pins 2.7 and 2.6 must be either open or pulled high to avoid placing the DS2250 in a parallel load cycle. Although an 11.0592 MHz crystal is shown in Figure 3, a variety of crystal frequencies and loader baud rates are supported, shown in Table 2. The serial loader is designed to operate across a three–wire interface from a standard UART. The receive, transmit, and ground wires are all that are necessary to establish communication with the DS2250.

The Serial Bootstrap Loader implements an easy–to–use command line interface which allows an application program in an Intel hex representation to be loaded into and read back from the device. Intel hex is the typical format which existing 8051 cross–assemblers output. The serial loader responds to single character commands which are summarized below:

COMMAND	FUNCTION
C	Return CRC–16 checksum of embedded RAM
D	Dump Intel Hex File
F	Fill embedded RAM block with constant
K	Load 40–bit Encryption Key
L	Load Intel Hex File
R	Read MCON register
T	Trace (Echo) incoming Intel Hex data
U	Clear Security Lock
V	Verify Embedded RAM with incoming Intel Hex
W	Write MCON register
Z	Set Security Lock
P	Put a value to a port.
G	Get a value from a port.

PARALLEL PROGRAM LOAD CYCLES Table 1

MODE	RST	$\overline{PSEN}$	$\overline{PROG}$	$\overline{EA}$	P2.7	P2.6	P2.5
Program	1	0	0	V_{PP}	1	0	X
Security Set	1	0	0	V_{PP}	1	1	X
Verify	1	X	X	1	0	0	X
Prog Expanded	1	0	0	V_{PP}	0	1	0
Verify Expanded	1	0	1	1	0	1	0
Prog MCON or Key registers	1	0	0	V_{PP}	0	1	1
Verify MCON registers	1	0	1	1	0	1	1

The Parallel Program Cycle is used to load a byte of data into a register or memory location within the DS2250. The Verify Cycle is used to read this byte back for comparison with the originally loaded value to verify proper load ing. The Security Set Cycle may be used to enable and the Software Security feature of the DS2250. One may also enter bytes for the MCON register or for the five encryption registers using the Program MCON cycle. When using this cycle, the absolute register address must be presented at Ports 1 and 2 as in the normal program cycle (Port 2 should be 00H). The MCON contents can likewise be verified using the Verify MCON cycle.

When the DS2250 first detects a Parallel Program Strobe pulse or a Security Set Strobe pulse while in the Program Load Mode following a Power On Reset, the internal hardware of the DS2250 is initialized so that an existing 4 Kbyte program can be programmed into a DS2250 with little or no modification. This initialization automatically sets the Range Address for 8 Kbytes and maps the lowest 4 Kbyte bank of Embedded RAM as program memory. The next 4 Kbytes of Embedded RAM are mapped as Data Memory.

In order to program more than 4 Kbytes of program code, the Program/Verify Expanded cycles can be used. Up to 32 Kbytes of program code can be entered and verified. Note that the expanded 32 Kbyte Program/Verify cycles take much longer than the normal 4 Kbyte Program/Verify cycles.

A typical parallel loading session would follow this procedure. First, set the contents of the MCON register with the correct range and partition only if using expanded programming cycles. Next, the encryption registers can be loaded to enable encryption of the program/data memory (not required). Then, program the DS2250 using either normal or expanded program cycles and check the memory contents using Verify cycles. The last operation would be to turn on the security lock feature by either a Security Set cycle or by explicitly writing to the MCON register and setting MCON.0 to a 1.

SERIAL LOADER BAUD RATES FOR DIFFERENT CRYSTAL FREQUENCIES Table 2

CRYSTAL FREQ (MHz)	BAUD RATE					
	300	1200	2400	9600	19200	57600
14.7456		Y	Y	Y	Y	
11.0592	Y	Y	Y	Y	Y	Y
9.21600	Y	Y	Y	Y		
7.37280	Y	Y	Y	Y		
5.52960	Y	Y	Y	Y		
1.84320	Y	Y	Y	Y		

ADDITIONAL INFORMATION

A complete description for all operational aspects of the DS2250, is provided in the User's Guide section of the Soft Microcontroller Data Book.

DEVELOPMENT SUPPORT

Dallas Semiconductor offers a kit package for developing and testing user code. The DS5000TK Evaluation Kit allows the user to download Intel hex formatted code directly to the DS2250 from a PC–XT/AT or compatible computer. The kit consists of a DS5000T–32–12, an interface pod, demo software, and an RS232 connector that attaches to the COM1 or COM2 serial port of a PC. The kit can be used with a DS2250. A mechanical adaptor, the DS9075–40V, allows a DS2250 to be used in the DS5000TK. See the User's Guide for further details.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

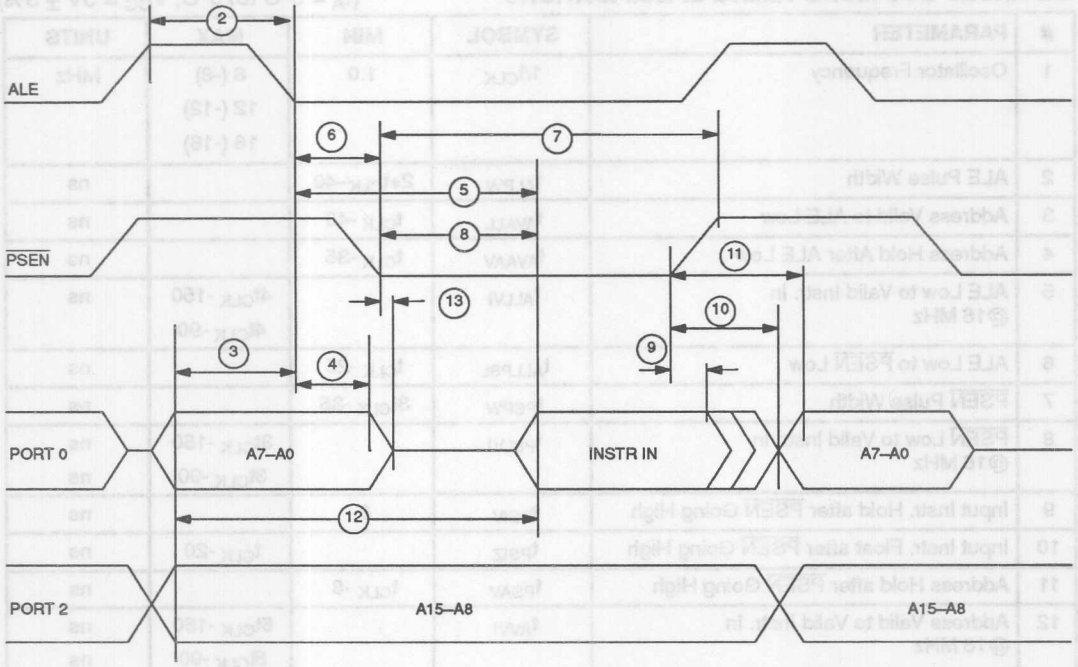
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	-0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC} +0.3	V	1
Input High Voltage RST, XTAL2	V_{IH2}	3.5		V_{CC} +0.3	V	1
Output Low Voltage @ I_{OL}=1.6mA (Ports 1, 2, 3)	V_{OL1}		.15	0.45	V	
Output Low Voltage @ I_{OL}=3.2mA (Ports 0, ALE, $\overline{PSEN}$)	V_{OL2}		.15	0.45	V	1
Output High Voltage @ I_{OH}=-80µA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @I_{OH}=-400 µA (Ports 0, ALE, $\overline{PSEN}$)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN} = 0.45V (Ports 1, 2, 3)	I_{IL}			-50	µA	
Transition Current; 1 to 0 V_{IN} = 2.0V (Ports 1, 2, 3)	I_{TL}			-500	µA	
Input Leakage Current 0.45 < V_{IN} < V_{CC} (Port 0)	I_L			±10	µA	
RST, $\overline{EA}$ Pulldown Resistor	R_{RE}	40		125	Kohm	
Stop Mode Current	I_{SM}			80	µA	4
Power Fail Warning Voltage	V_{PFW}	4.15	4.6	4.75	V	1
Minimum Operating Voltage	V_{CCmin}	4.05	4.5	4.65	V	1
Programming Supply Voltage (Parallel Program Mode)	V_{PP}	12.5		13	V	1
Program Supply Current	I_{PP}		15	20	mA	
Operating Current DS2250–8K DS2250–32K @ 12 MHz DS2250T–64–16 @ 16 MHz	I_{CC}			43 48 54	mA	2
Idle Mode Current	I_{CC}			6.2	mA	3

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

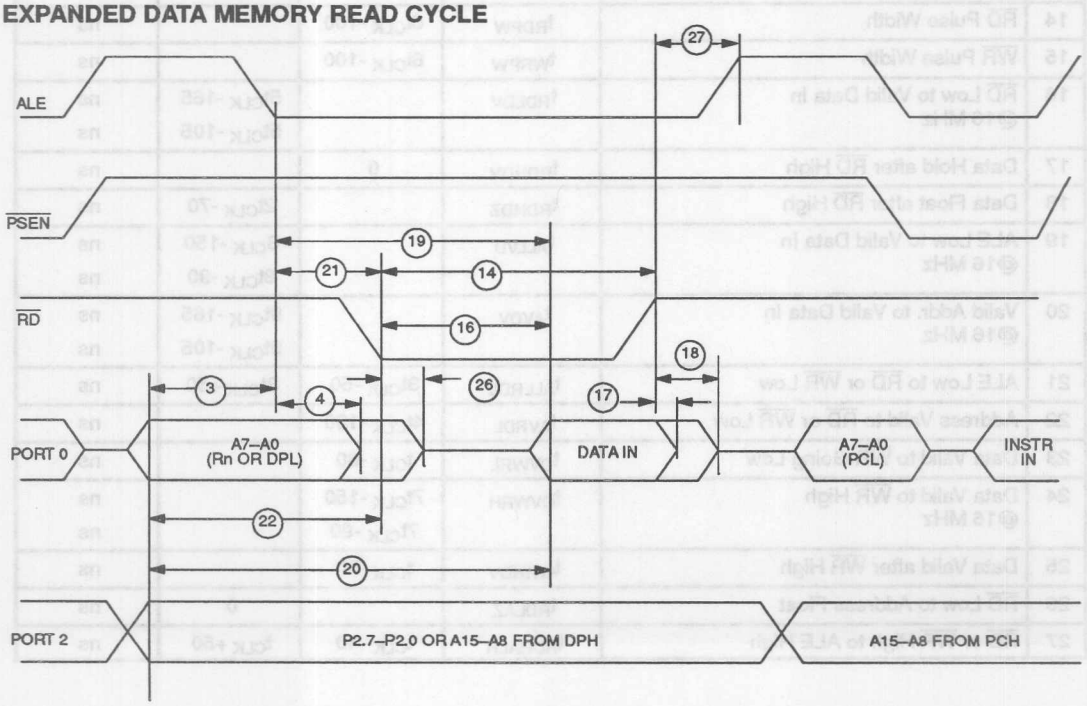
(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	8 (-8) 12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2*t_{CLK}-40$		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK} -40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK} -35		ns
5	ALE Low to Valid Instr. In @16 MHz	t_{ALLVI}		$4t_{CLK}$ -150 $4t_{CLK}$ -90	ns
6	ALE Low to $\overline{PSEN}$ Low	t_{ALLPSL}	t_{CLK} -25		ns
7	$\overline{PSEN}$ Pulse Width	t_{PSPW}	$3t_{CLK}$ -35		ns
8	$\overline{PSEN}$ Low to Valid Instr. In @16 MHz	t_{PSLVI}		$3t_{CLK}$ -150 $3t_{CLK}$ -90	ns ns
9	Input Instr. Hold after $\overline{PSEN}$ Going High	t_{PSIV}	0		ns
10	Input Instr. Float after $\overline{PSEN}$ Going High	t_{PSIZ}		t_{CLK} -20	ns
11	Address Hold after $\overline{PSEN}$ Going High	t_{PSAV}	t_{CLK} -8		ns
12	Address Valid to Valid Instr. In @16 MHz	t_{AVVI}		$5t_{CLK}$ -150 $5t_{CLK}$ -90	ns ns
13	$\overline{PSEN}$ Low to Address Float	t_{PSLAZ}	0		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$ -100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$ -100		ns
16	$\overline{RD}$ Low to Valid Data In @16 MHz	t_{RDLDV}		$5t_{CLK}$ -165 $5t_{CLK}$ -105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$ -70	ns
19	ALE Low to Valid Data In @16 MHz	t_{ALLVD}		8_{CLK} -150 $8t_{CLK}$ -90	ns ns
20	Valid Addr. to Valid Data In @16 MHz	t_{AVDV}		$9t_{CLK}$ -165 $9t_{CLK}$ -105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$ -50	$3t_{CLK}$ +50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$ -130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK} -60		ns
24	Data Valid to $\overline{WR}$ High @16 MHz	t_{DVWRH}	$7t_{CLK}$ -150 $7t_{CLK}$ -90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK} -50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK} -40	t_{CLK} +50	ns

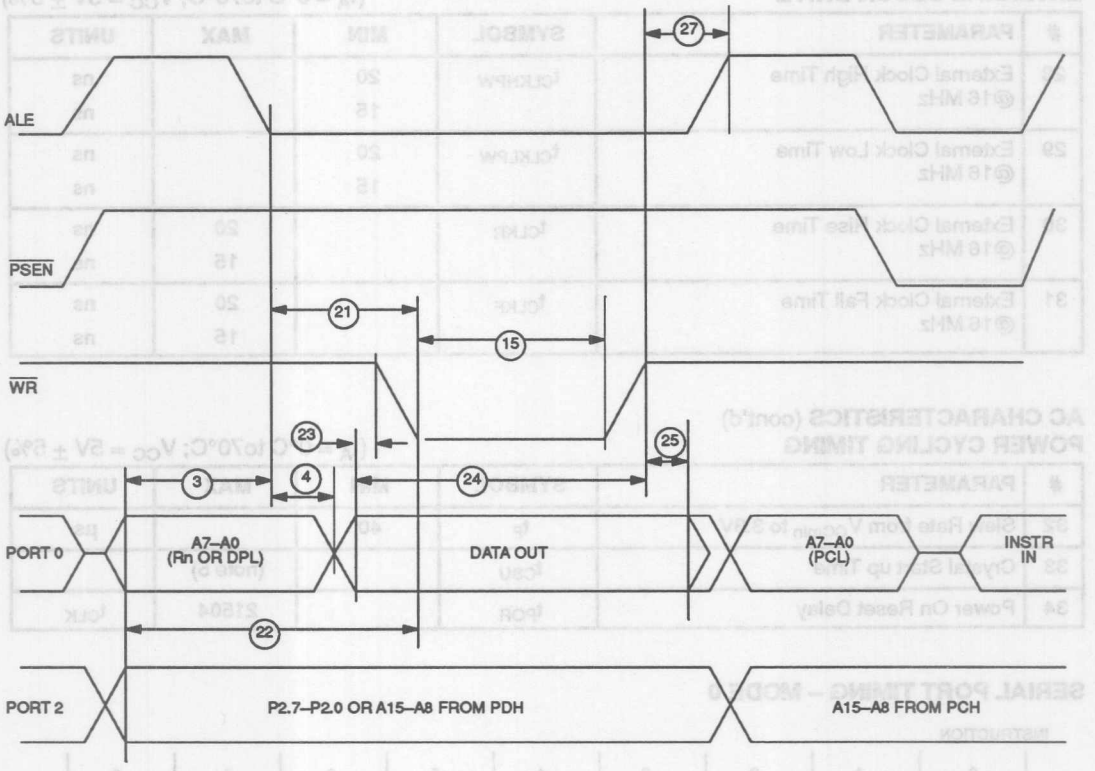
EXPANDED PROGRAM MEMORY READ CYCLE



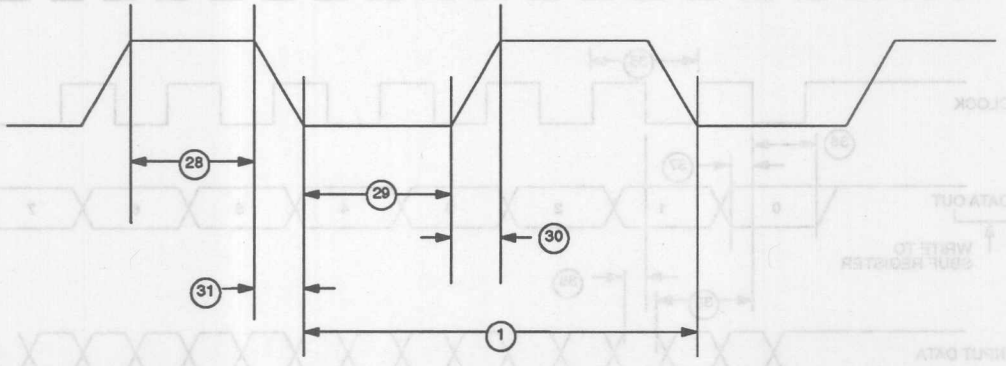
EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE



EXTERNAL CLOCK TIMING



AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

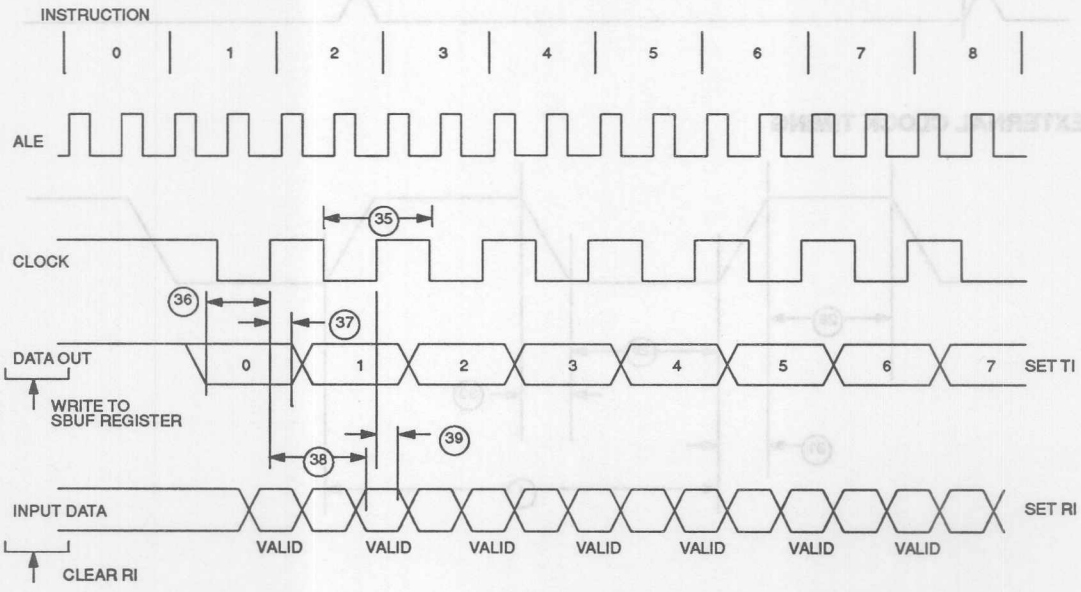
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @16 MHz	t_{CLKF}		20 15	ns ns

AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCmin} to 3.3V	t_F	40		µs
33	Crystal Start up Time	t_{CSU}		(note 5)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

SERIAL PORT TIMING – MODE 0

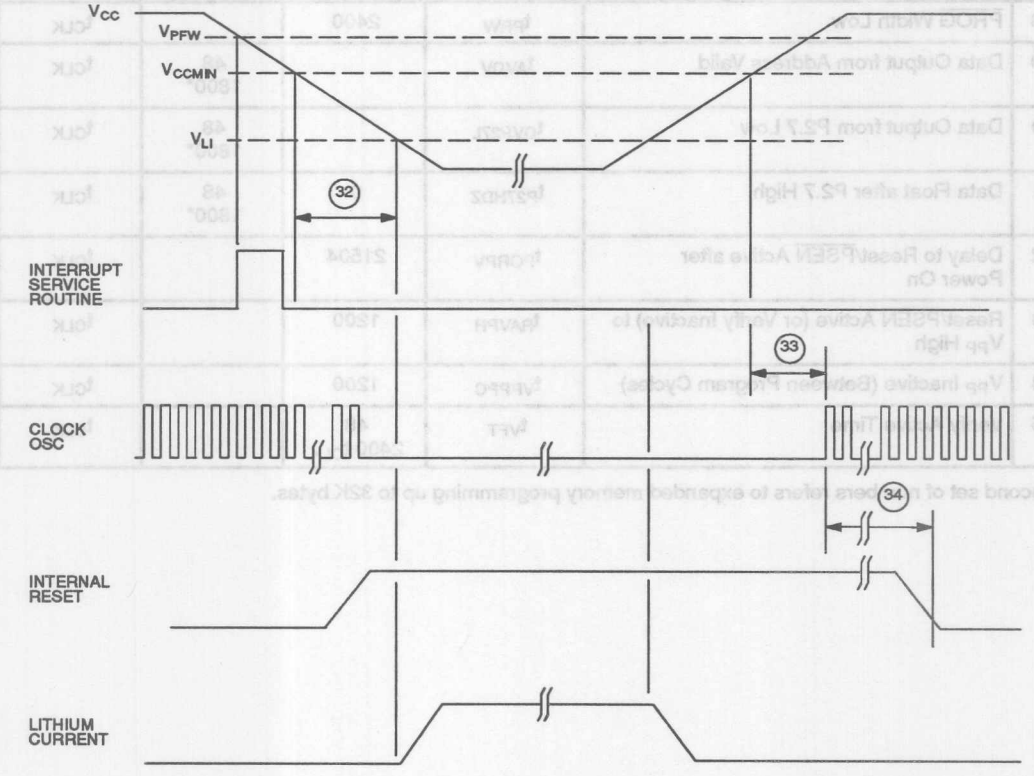


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$ -133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$ -117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$ -133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

POWER CYCLE TIMING



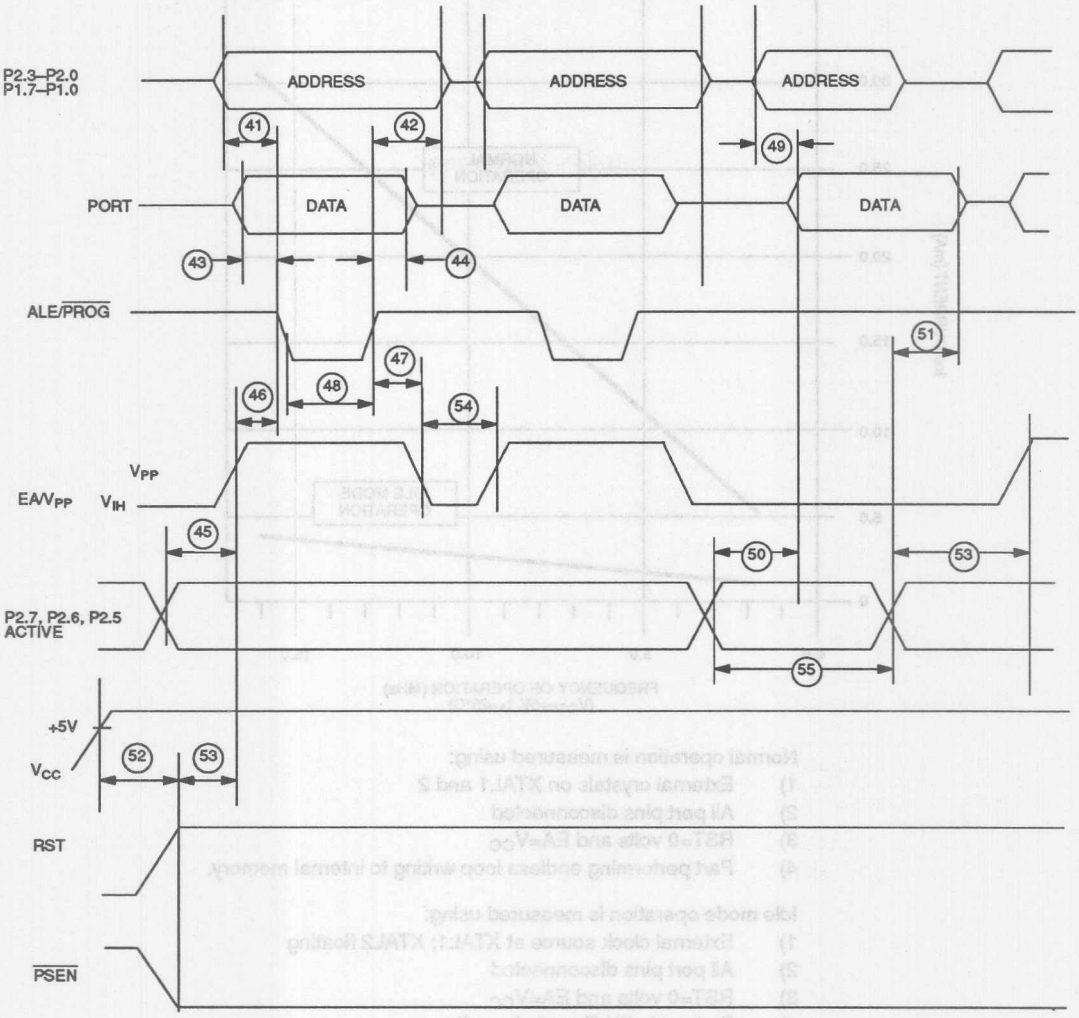
AC CHARACTERISTICS (cont'd)
PARALLEL PROGRAM LOAD TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Oscillator Frequency	$1/t_{CLK}$	1.0	12.0	MHz
41	Address Setup to $\overline{PROG}$ Low	t_{AVPRL}	0		
42	Address Hold after $\overline{PROG}$ High	t_{PRHAV}	0		
43	Data Setup to $\overline{PROG}$ Low	t_{DVPRL}	0		
44	Data Hold after $\overline{PROG}$ High	t_{PRHDV}	0		
45	P2.7, 2.6, 2.5 Setup to V_{PP}	t_{P27HVP}	0		
46	V_{PP} Setup to $\overline{PROG}$ Low	t_{VPHPRL}	0		
47	V_{PP} Hold after $\overline{PROG}$ Low	t_{PRHVPL}	0		
48	$\overline{PROG}$ Width Low	t_{PRW}	2400		t_{CLK}
49	Data Output from Address Valid	t_{AVDV}		48 1800*	t_{CLK}
50	Data Output from P2.7 Low	t_{DVP27L}		48 1800*	t_{CLK}
51	Data Float after P2.7 High	t_{P27HDZ}	0	48 1800*	t_{CLK}
52	Delay to Reset/$\overline{PSEN}$ Active after Power On	t_{PORPV}	21504		t_{CLK}
53	Reset/$\overline{PSEN}$ Active (or Verify Inactive) to V_{PP} High	t_{RAVPH}	1200		t_{CLK}
54	V_{PP} Inactive (Between Program Cycles)	t_{VPPPC}	1200		t_{CLK}
55	Verify Active Time	t_{VFT}	48 2400 t_{CLK}		t_{CLK}

* Second set of numbers refers to expanded memory programming up to 32K bytes.

PARALLEL PROGRAM LOAD TIMING

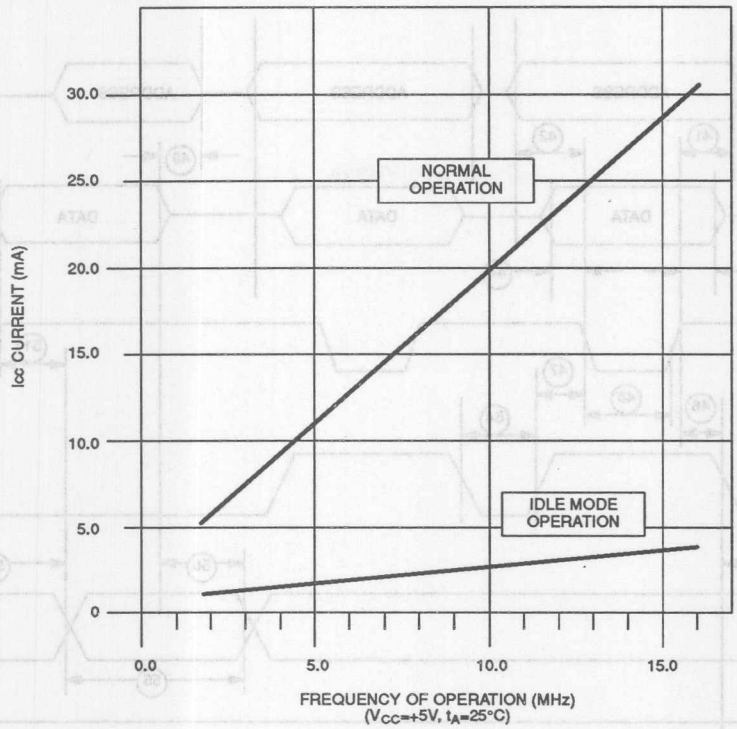


CAPACITANCE

(test frequency = 1 MHz; t_A = 25°C)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Output Capacitance	C_O			10	pF	
Input Capacitance	C_I			10	pF	

DS2250 TYPICAL I_{CC} VS. FREQUENCY



Normal operation is measured using:

1) External crystals on XTAL1 and 2
2) All port pins disconnected
3) RST=0 volts and EA=V_{CC}
4) Part performing endless loop writing to internal memory.

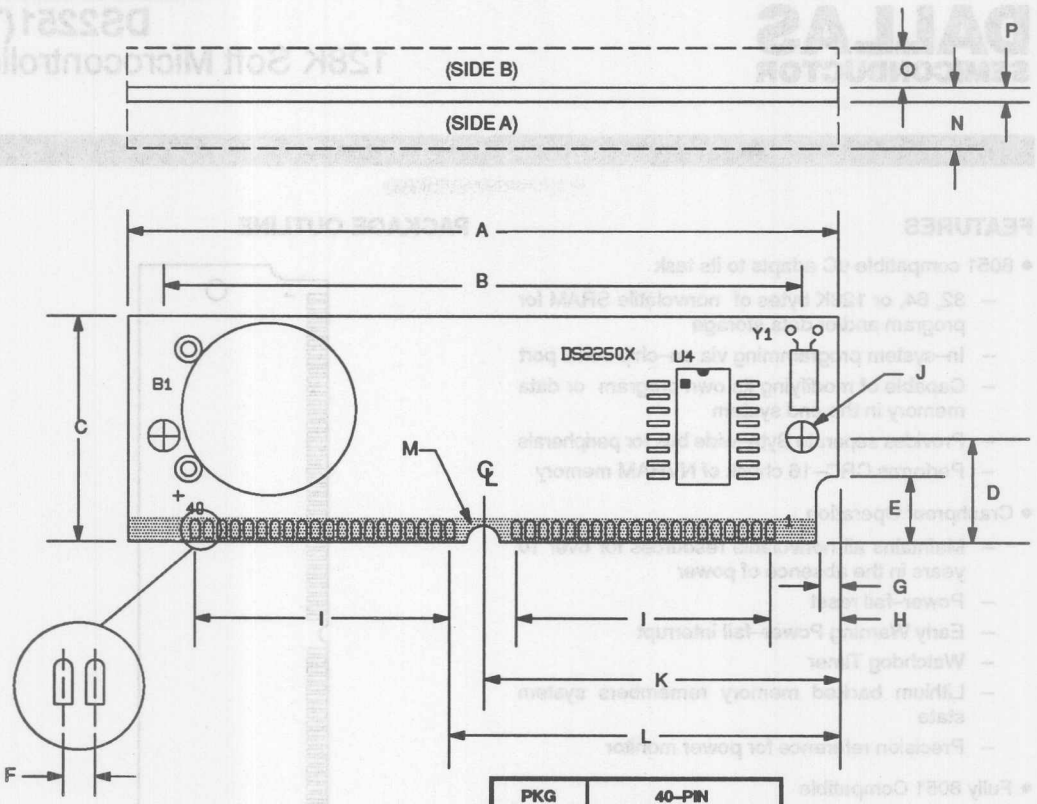
Idle mode operation is measured using:

1) External clock source at XTAL1; XTAL2 floating
2) All port pins disconnected
3) RST=0 volts and EA=V_{CC}
4) Part set in IDLE mode by software.

NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with tCLKR, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = RST = PORT0 = V_{CC}.
3. Idle mode I_{CC} is measured with all output pins disconnected; XTAL1 driven at 8 MHz with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = PORT0 = V_{CC}, RST = V_{SS}.
4. Stop mode I_{CC} is measured with all output pins disconnected; $\overline{EA}$ = PORT0 = V_{CC}; XTAL2 not connected; RST = V_{SS}.
5. Crystal start up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on-chip oscillator. The user should check with the crystal vendor for the worst case spec on this time.

PACKAGE DRAWING



PKG	40-PIN	
DIM	MIN	MAX
A	2.645	2.655
B	2.379	2.389
C	0.845	0.855
D	0.395	0.405
E	0.245	0.255
F	0.050 BSC	
G	0.075	0.085
H	0.245	0.255
I	0.950 BSC	
J	0.120	0.130
K	1.320	1.330
L	1.445	1.455
M	0.057	0.067
N		0.160
O		0.195
P		0.054

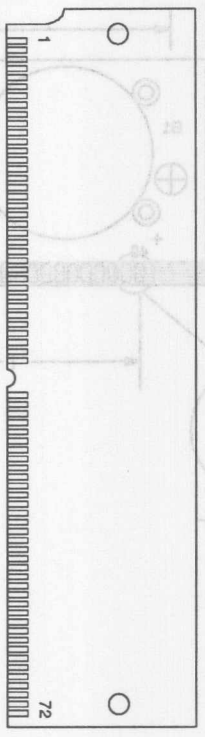
DALLAS SEMICONDUCTOR

DS2251(T)
128K Soft Microcontroller

FEATURES

- 8051 compatible uC adapts to its task
  - 32, 64, or 128K bytes of nonvolatile SRAM for program and/or data storage
  - In–system programming via on–chip serial port
  - Capable of modifying its own program or data memory in the end system
  - Provides separate Byte–wide bus for peripherals
  - Performs CRC–16 check of NVRAM memory
- Crashproof Operation
  - Maintains all nonvolatile resources for over 10 years in the absence of power
  - Power–fail reset
  - Early Warning Power–fail Interrupt
  - Watchdog Timer
  - Lithium backed memory remembers system state
  - Precision reference for power monitor
- Fully 8051 Compatible
  - 128 bytes scratchpad RAM
  - Two timer/counters
  - On–chip serial port
  - 32 parallel I/O port pins
- Optional permanently powered Real–time Clock (DS2251T)

PACKAGE OUTLINE



72–PIN SIMM

DESCRIPTION

The DS2251(T) is an 8051 compatible microcontroller based on nonvolatile RAM technology. It is designed for systems that need large quantities of nonvolatile memory. Like other members of the Soft Micro family, it provides full compatibility with the 8051 instruction set, timers, serial port, and parallel I/O ports. By using NVRAM instead of ROM, the user can program, then reprogram the microcontroller while in–system. The application software can even change its own operation. This allows frequent software upgrades, adaptive programs, customized systems, etc. In addition, by using NVSRAM, the DS2251(T) is ideal for data logging applications. The DS2251T provides a powerful Real–time Clock with interrupts for time stamp and date. It keeps time to one hundredth of second using its on–board 32 KHz crystal.

The DS2251(T) provides the benefits of NVRAM without using I/O resources. Between 32K bytes and 128K bytes of on–board NVRAM are available. A non–multiplexed Byte–wide address and data bus is used for memory access. This bus, which is available at the connector, can perform all memory access and also provides decoded chip enables for off–board memory

mapped peripherals. This leaves the 32 I/O port pins free for application use.

The DS2251(T) provides crashproof operation in portable systems or systems with unreliable power. These features include the ability to save the operating state, Power–fail Reset, Power–fail Interrupt, and Watchdog Timer. All nonvolatile memory and resources are maintained for over 10 years at room temperature in the absence of power.

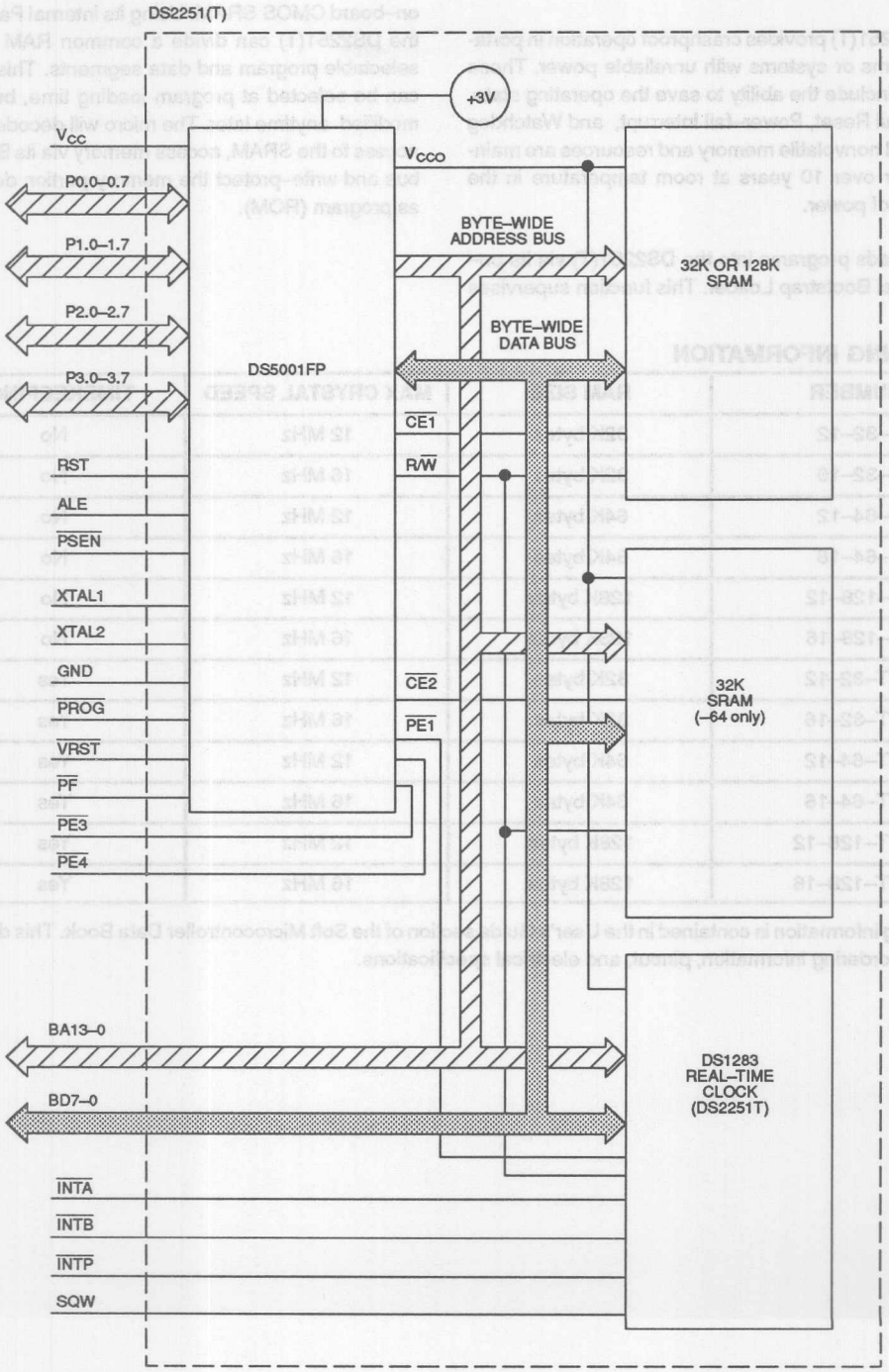
A user loads programs into the DS2251(T) via its on–chip Serial Bootstrap Loader. This function supervises the loading of software into NVRAM, validates it, then becomes transparent to the user. Software is stored in on–board CMOS SRAM. Using its internal Partitioning, the DS2251(T) can divide a common RAM into user selectable program and data segments. This Partition can be selected at program loading time, but can be modified anytime later. The micro will decode memory access to the SRAM, access memory via its Byte–wide bus and write–protect the memory portion designated as program (ROM).

ORDERING INFORMATION

PART NUMBER	RAM SIZE	MAX CRYSTAL SPEED	TIMEKEEPING?
DS2251–32–12	32K bytes	12 MHz	No
DS2251–32–16	32K bytes	16 MHz	No
DS2251–64–12	64K bytes	12 MHz	No
DS2251–64–16	64K bytes	16 MHz	No
DS2251–128–12	128K bytes	12 MHz	No
DS2251–128–16	128K bytes	16 MHz	No
DS2251T–32–12	32K bytes	12 MHz	Yes
DS2251T–32–16	32K bytes	16 MHz	Yes
DS2251T–64–12	64K bytes	12 MHz	Yes
DS2251T–64–16	64K bytes	16 MHz	Yes
DS2251T–128–12	128K bytes	12 MHz	Yes
DS2251T–128–16	128K bytes	16 MHz	Yes

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pinout, and electrical specifications.

DS2251(T) BLOCK DIAGRAM Figure 1



PIN ASSIGNMENT

1	P1.0	19	XTAL2	37	P0.2	55	$\overline{INTB}$
2	P1.1	20	GND	38	P0.1	56	BD0
3	P1.2	21	P2.0	39	P0.0	57	BD1
4	P1.3	22	P2.1	40	V_{CC}	58	BD2
5	P1.4	23	P2.2	41	BA0	59	BD3
6	P1.5	24	P2.3	42	BA1	60	BD4
7	P1.6	25	P2.4	43	BA2	61	BD5
8	P1.7	26	P2.5	44	BA3	62	BD6
9	RST	27	P2.6	45	BA4	63	BD7
10	P3.0 RXD	28	P2.7	46	BA5	64	R/$\overline{W}$
11	P3.1 TXD	29	$\overline{PSEN}$	47	BA6	65	$\overline{PF}$
12	P3.2 $\overline{INT0}$	30	ALE	48	BA7	66	$\overline{PE3}$
13	P3.3 $\overline{INT1}$	31	$\overline{PROG}$	49	BA8	67	$\overline{PE4}$
14	P3.4 T0	32	P0.7	50	BA9	68	$\overline{INTP}$
15	P3.5 T1	33	P0.6	51	BA10	69	$\overline{INTA}$
16	P3.6 $\overline{WR}$	34	P0.5	52	BA11	70	SQW
17	P3.7 $\overline{RD}$	35	P0.4	53	BA12	71	$\overline{VRST}$
18	XTAL1	36	P0.3	54	BA13	72	BA15

PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
39–32	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull–ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
1–8	P1.0 – P1.7 General purpose I/O Port 1.
21–28	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.
10	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
11	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
12	P3.2 $\overline{INT0}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
13	P3.3 $\overline{INT1}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
14	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
15	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.

PIN NUMBER	DESCRIPTION
16	P3.6 $\overline{WR}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
17	P3.7 $\overline{RD}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
9	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally, can be left unconnected if not used. An RC power–on reset circuit is not needed and is NOT recommended.
29	$\overline{PSEN}$ Program Store Enable. This active low signal is used to enable an external program memory when using the Expanded bus. It is normally an output and should be unconnected if not used.
30	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch.
19, 18	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
20	GND Logic ground.
40	V_{CC} +5V
72	BA15 Monitor test point to reflect the logical value of A15. Not needed for memory access.
54–41	BA13–0 Byte–wide Address bus bits 13–0. This bus is combined with the non–multiplexed data bus (BD7–0) to access on–board NVSRAM and off–board peripherals. Peripheral decoding is performed using $\overline{PE3}$ and $\overline{PE4}$. These are on 16K boundaries, so BA14 or BA15 are not needed. Read/write access is controlled by R/$\overline{W}$. BA13–0 connect directly to memory mapped peripherals.
63–56	BD7–0 Byte–wide Data bus bits 7–0. This 8 bit bi–directional bus is combined with the non–multiplexed address bus (BA14–0) to access on–board NVSRAM and off–board peripherals.
64	R/$\overline{W}$ Read/Write. This signal provides the write enable to the SRAMs on the Byte–wide bus. It is controlled by the memory map and Partition. The blocks selected as Program (ROM) will be write protected. This signal is also used for the write enable to off–board peripherals.
66	$\overline{PE3}$ Peripheral Enable 3. Accesses data memory between addresses 8000h and BFFFh when the PES bit is set to a logic 1. $\overline{PE3}$ is not lithium backed and can be connected to any type of peripheral function.
67	$\overline{PE4}$ Peripheral Enable 4. Accesses data memory between addresses C000h and FFFFh when the PES bit is set to a logic 1. $\overline{PE4}$ is not lithium backed and can be connected to any type of peripheral function.

PIN NUMBER	DESCRIPTION
31	$\overline{PROG}$ Invokes the Bootstrap loader on a falling edge. This signal should be debounced so that only one edge is detected. If connected to ground, the micro will enter Bootstrap loading on power up. This signal is pulled up internally.
71	$\overline{VRST}$ This I/O pin indicates that the power supply (V_{CC}) has fallen below the V_{CCMIN} level and the micro is in a reset state. When this occurs, the DS2251(T) will drive this pin to a logic 0. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V. Because it is an I/O pin, it will also force a reset if pulled low externally. This allows multiple parts to synchronize their power–down resets.
65	$\overline{PF}$ This output goes to a logic 0 to indicate that the micro has switched to lithium backup. It corresponds to $V_{CC} < V_{LI}$. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V.
55	$\overline{INTB}$ $\overline{INTB}$ from the Real–time Clock. This output may be connected to a micro interrupt input.
68	$\overline{INTP}$ $\overline{INTP}$ from the Real–time Clock. This open–drain output requires a pull–up and may beconnected to a micro interrupt input.
69	$\overline{INTA}$ $\overline{INTA}$ from the Real–time Clock. This output may be connected to a micro interrupt input.
70	SQW SQW output from the DS1283 Real–time Clock. Can be programmed to output an 1024 Hz square wave.

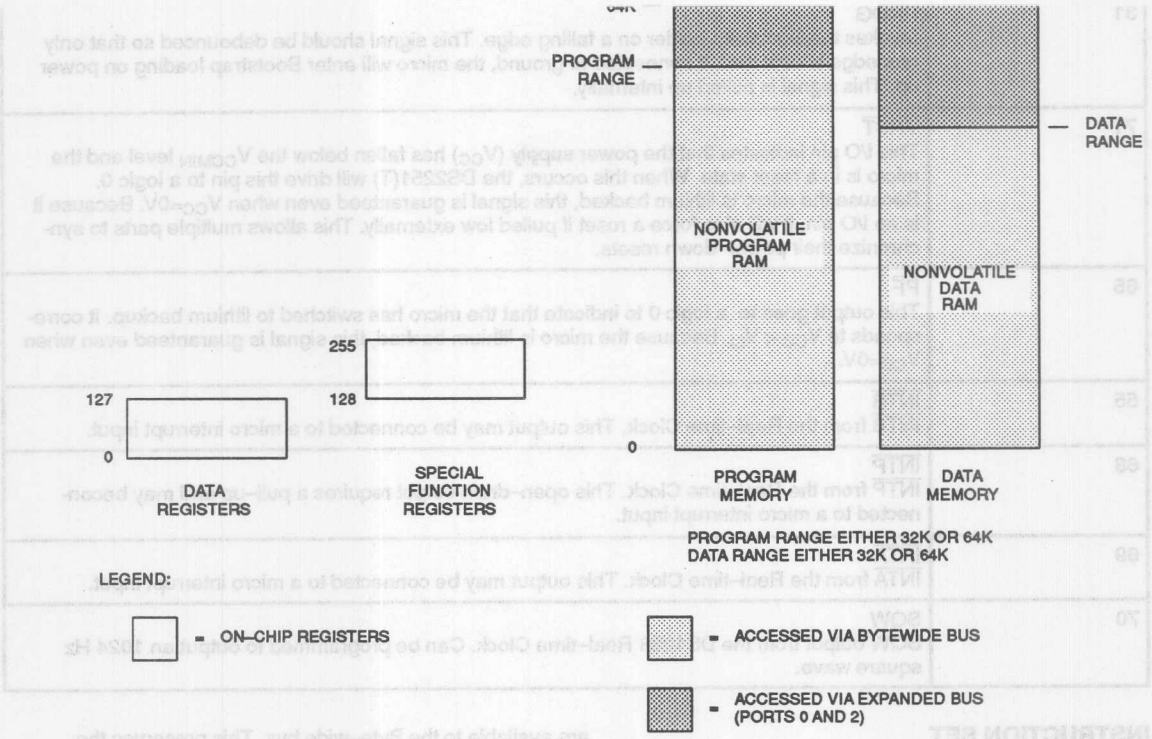
INSTRUCTION SET

The DS2251(T) executes an instruction set that is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages such as assemblers and compilers that have been written for the 8051 are compatible with the DS2251(T).

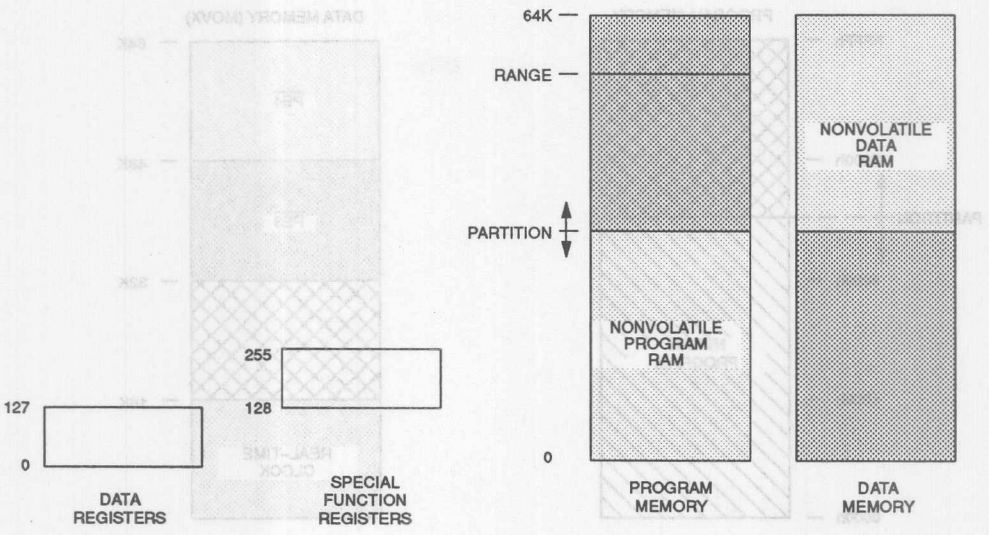
A complete description of the instruction set and operation are provided in the User's Guide section of the Soft Microcontroller Data Book.

MEMORY ORGANIZATION

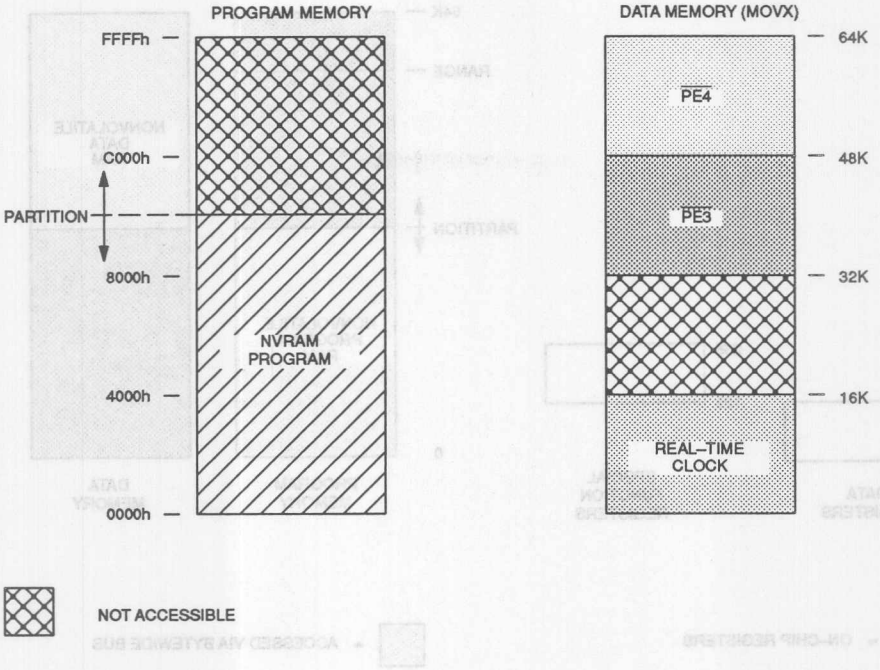
Figure 2 illustrates the memory map accessed by the DS2251(T). The entire 64K of program and 64K of data are available to the Byte–wide bus. This preserves the I/O ports for application use. The user controls the portion of memory that is actually mapped to the Byte–wide bus by selecting the Program Range and Data Range. Any area not mapped into the NVRAM is reached via the Expanded bus on Ports 0 and 2. An alternate configuration allows dynamic Partitioning of a 64K space as shown in Figure 3. Selecting PES=1 provides access to the Real–time Clock on the DS2251T and enables $\overline{PE3}$ and $\overline{PE4}$ for peripheral access as shown in Figure 4. These selections are made using Special Function Registers. The memory map and its controls are covered in detail in the User's Guide section of the Soft Microcontroller Data Book.


PROGRAM RANGE
DATA RANGE
NONVOLATILE PROGRAM RAM
NONVOLATILE DATA RAM
255
128
127
0
0
DATA REGISTERS
SPECIAL FUNCTION REGISTERS
PROGRAM MEMORY
DATA MEMORY
PROGRAM RANGE EITHER 32K OR 64K
DATA RANGE EITHER 32K OR 64K
LEGEND:
= ON–CHIP REGISTERS
= ACCESSED VIA BYTEWIDE BUS
= ACCESSED VIA EXPANDED BUS (PORTS 0 AND 2)

MEMORY MAP OF THE DS2251(T) WITH PM=0 Figure 3



MEMORY MAP OF THE DS2251(T) WITH PES=1 Figure 4



POWER MANAGEMENT

The DS2251(T) monitors V_{CC} to provide Power–fail Reset, early warning Power–fail Interrupt, and switch over to lithium backup. It uses an internal band–gap reference in determining the switch points. These are called V_{PFW}, V_{CCMIN}, and V_{LI} respectively. When V_{CC} drops below V_{PFW}, the DS2251(T) will perform an interrupt vector to location 2Bh if the power fail warning was enabled. Full processor operation continues regardless. When power falls further to V_{CCMIN}, the DS2251(T) invokes a reset state. No further code execution will be performed unless power rises back above V_{CCMIN}. All decoded chip enables and the R/$\overline{W}$ signal go to an inactive (logic 1) state. The $\overline{VRST}$ signal will be driven to a a logic 0. V_{CC} is still the power source at this time. When V_{CC} drops further to below V_{LI}, internal circuitry will switch to the built–in lithium cell for power. The majority of internal circuits will be disabled and the remaining nonvolatile states will be retained. $\overline{PF}$ will be driven to a logic 0. The User's Guide has more information on this topic. The trip points V_{CCMIN} and V_{PFW} are listed in the electrical specifications.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

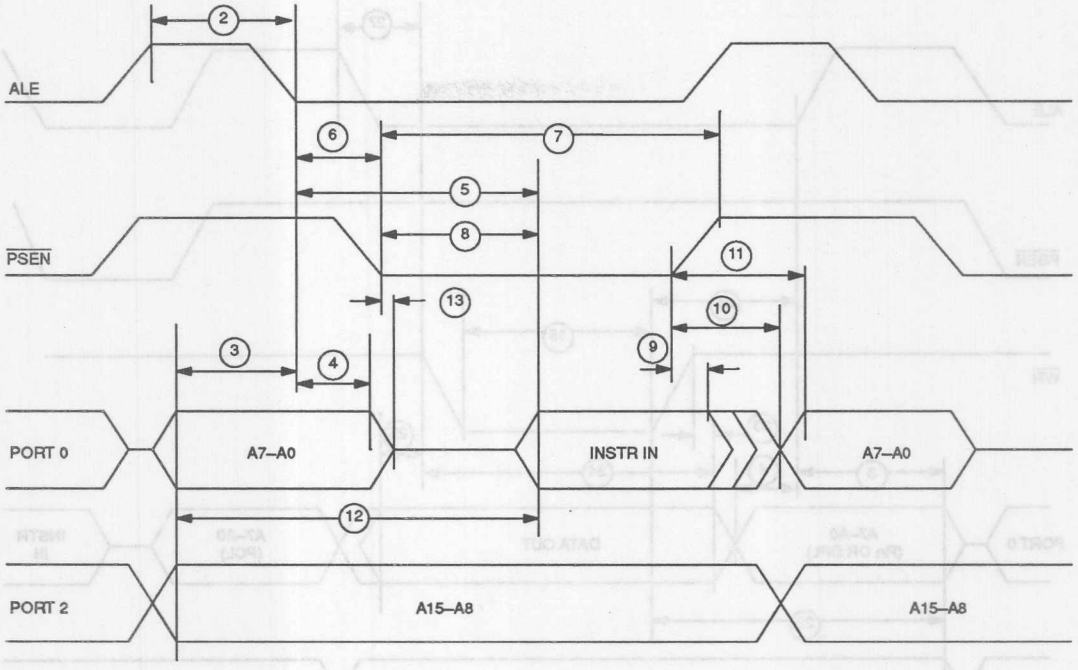
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	-0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC} +0.3	V	1
Input High Voltage RST, XTAL1 $\overline{PROG}$	V_{IH2}	3.5		V_{CC} +0.3	V	1
Output Low Voltage @ I_{OL}=1.6mA (Ports 1, 2, 3)	V_{OL1}		0.15	0.45	V	1
Output Low Voltage @ I_{OL}=3.2mA (Ports 0, ALE, $\overline{PSEN}$, $\overline{PF}$, BA13–0, BD7–0, R/$\overline{W}$, $\overline{PE}$3–4)	V_{OL2}		0.15	0.45	V	1
Output High Voltage @ I_{OH}=-80µA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @ I_{OH}=-400 µA (Ports 0, ALE, $\overline{PSEN}$, $\overline{PF}$, BA13–0, BD7–0, R/$\overline{W}$, $\overline{PE}$3–4)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN} = 0.45V (Ports 1, 2, 3)	I_{IL}			-50	µA	
Transition Current; 1 to 0 V_{IN} = 2.0V (Ports 1, 2, 3)	I_{TL}			-500	µA	
Input Leakage Current 0.45 < V_{IN} < V_{CC} (Port 0)	I_{IL}			±10	µA	
RST Pulldown Resistor	R_{RE}	40		150	KΩ	
$\overline{VRST}$ Pullup Resistor	R_{VR}		4.7		KΩ	
$\overline{PROG}$ Pullup Resistor	R_{PR}		40		KΩ	
Power Fail Warning Voltage	V_{PFW}	4.25	4.37	4.50	V	1
Minimum Operating Voltage	V_{CCmin}	4.00	4.12	4.25	V	1
Operating Current	I_{CC}			45	mA	2
Idle Mode Current	I_{IDLE}			7.0	mA	3
Stop Mode Current	I_{STOP}			80	µA	4
Pin Capacitance	C_{IN}			10	pF	5
Reset Trip Point in Stop Mode w/BAT=3.0V w/BAT=3.3V		 4.0 4.4		 4.25 4.65	V	1

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

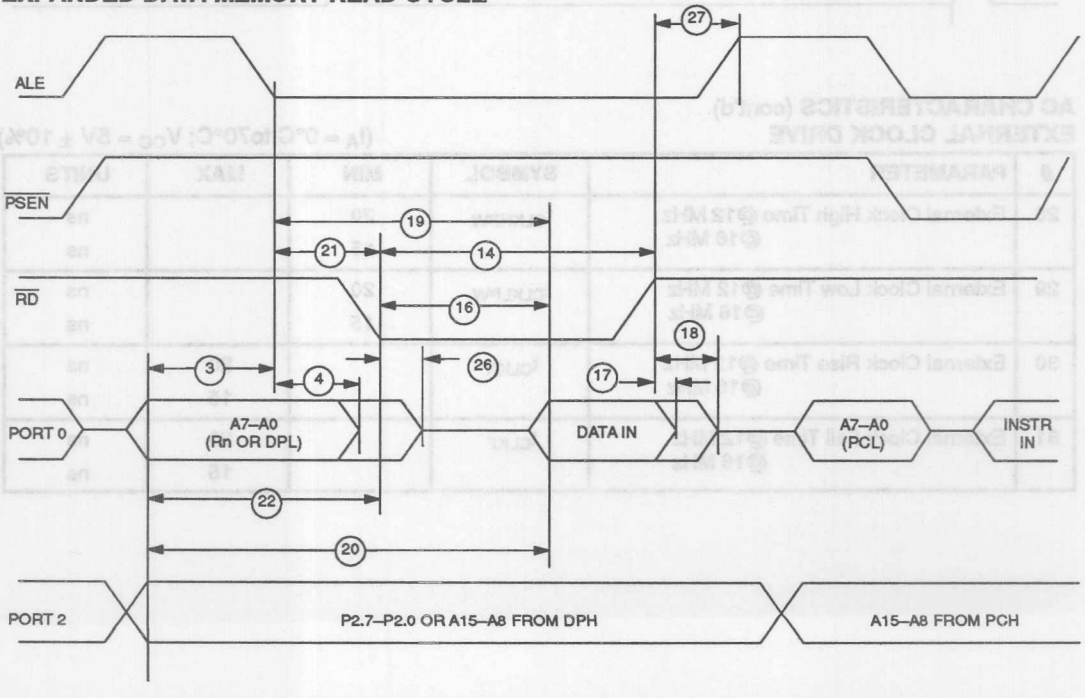
(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2t_{CLK}$–40		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK}-40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK}-35		ns
5	ALE Low to Valid Instr. In @12 MHz @16 MHz	t_{ALLVI}		$4t_{CLK}$-150 $4t_{CLK}$-90	ns
6	ALE Low to $\overline{PSEN}$ Low	t_{ALLPSL}	t_{CLK}-25		ns
7	$\overline{PSEN}$ Pulse Width	t_{PSPW}	$3t_{CLK}$-35		ns
8	$\overline{PSEN}$ Low to Valid Instr. In @12 MHz @16 MHz	t_{PSLVI}		$3t_{CLK}$-150 $3t_{CLK}$-90	ns ns
9	Input Instr. Hold after $\overline{PSEN}$ Going High	t_{PSIV}	0		ns
10	Input Instr. Float after $\overline{PSEN}$ Going High	t_{PSIX}		t_{CLK}-20	ns
11	Address Hold after $\overline{PSEN}$ Going High	t_{PSAV}	t_{CLK}-8		ns
12	Address Valid to Valid Instr. In @12 MHz @16 MHz	t_{AVVI}		$5t_{CLK}$-150 $5t_{CLK}$-90	ns ns
13	$\overline{PSEN}$ Low to Address Float	t_{PSLAZ}	0		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$-100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$-100		ns
16	$\overline{RD}$ Low to Valid Data In @12 MHz @16 MHz	t_{RDLDV}		$5t_{CLK}$-165 $5t_{CLK}$-105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$-70	ns
19	ALE Low to Valid Data In @12 MHz @16 MHz	t_{ALLVD}		$8t_{CLK}$-150 $8t_{CLK}$-90	ns ns
20	Valid Addr. to Valid Data In @12 MHz @16 MHz	t_{AVDV}		$9t_{CLK}$-165 $9t_{CLK}$-105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$-50	$3t_{CLK}$+50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$-130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK}-60		ns
24	Data Valid to $\overline{WR}$ High @12 MHz @16 MHz	t_{DVWRH}	$7t_{CLK}$-150 $7t_{CLK}$-90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK}-50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK}-40	t_{CLK}+50	ns

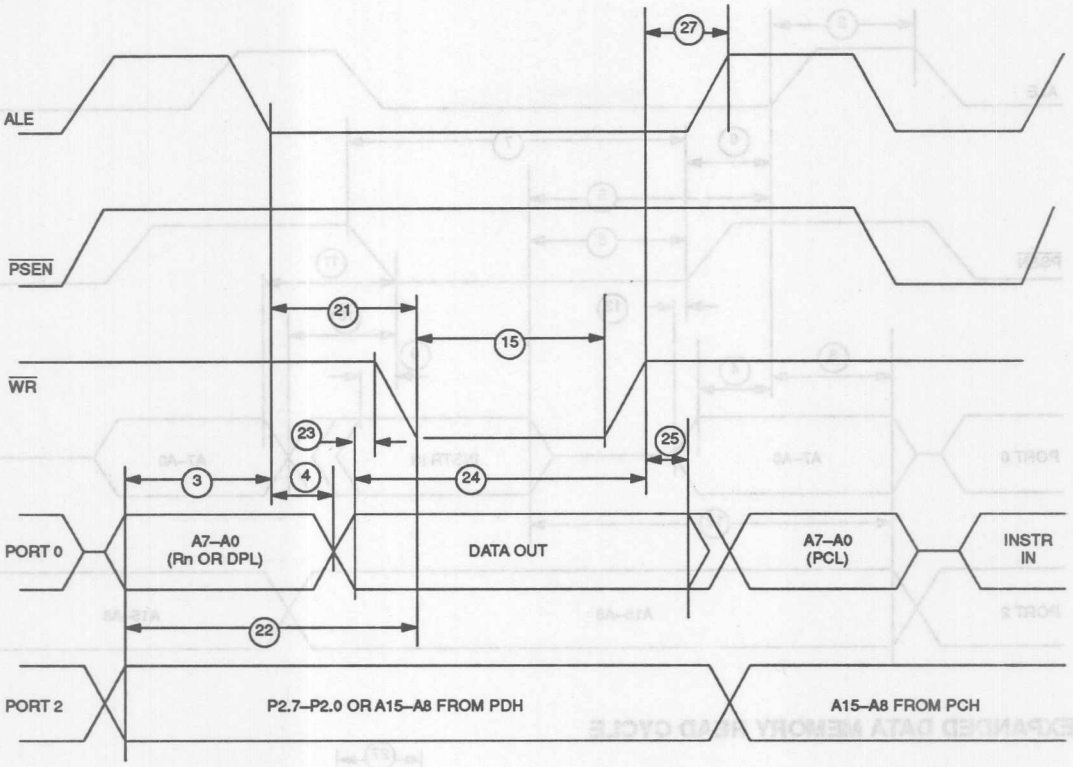
EXPANDED PROGRAM MEMORY READ CYCLE



EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE

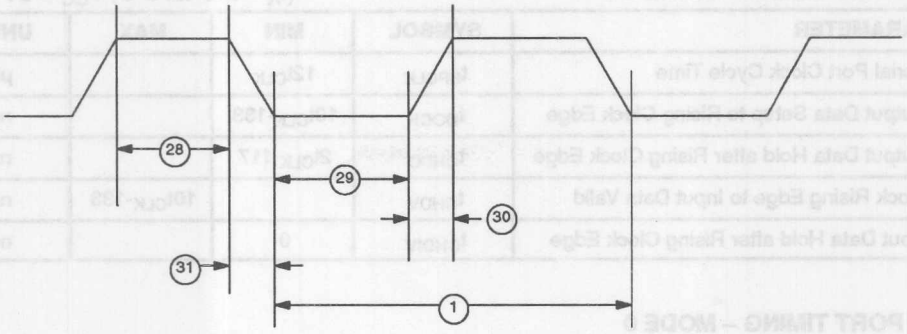


AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @12 MHz @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @12 MHz @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @12 MHz @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @12 MHz @16 MHz	t_{CLKF}		20 15	ns ns

EXTERNAL CLOCK TIMING

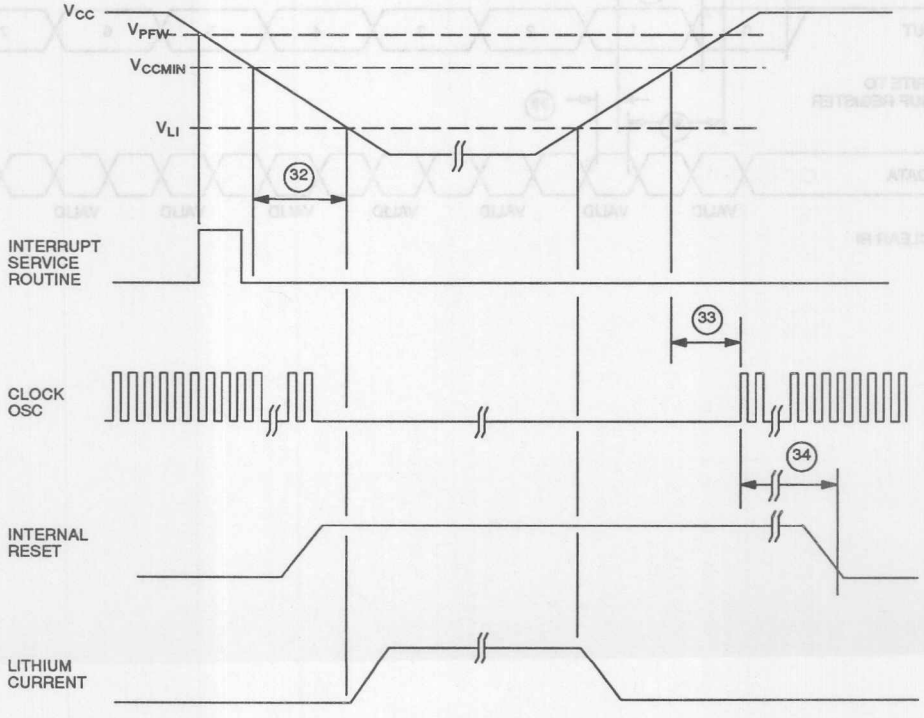


AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCMIN} to V_{LI}	t_F	130		µs
33	Crystal Start up Time	t_{CSU}		(note 6)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

POWER CYCLE TIMING

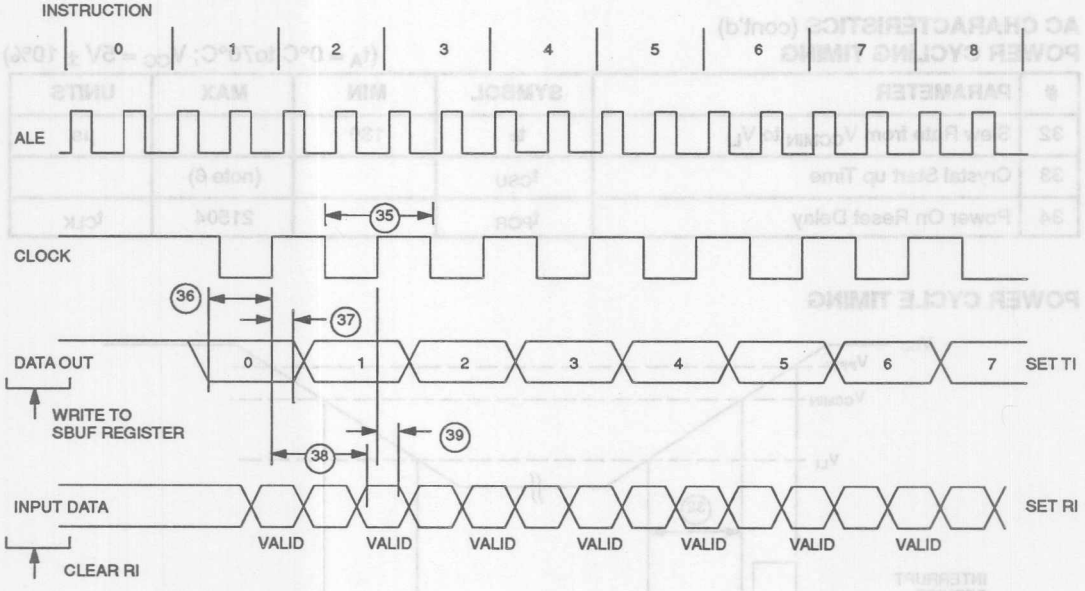


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Clock Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$-133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$-117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$-133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

SERIAL PORT TIMING – MODE 0

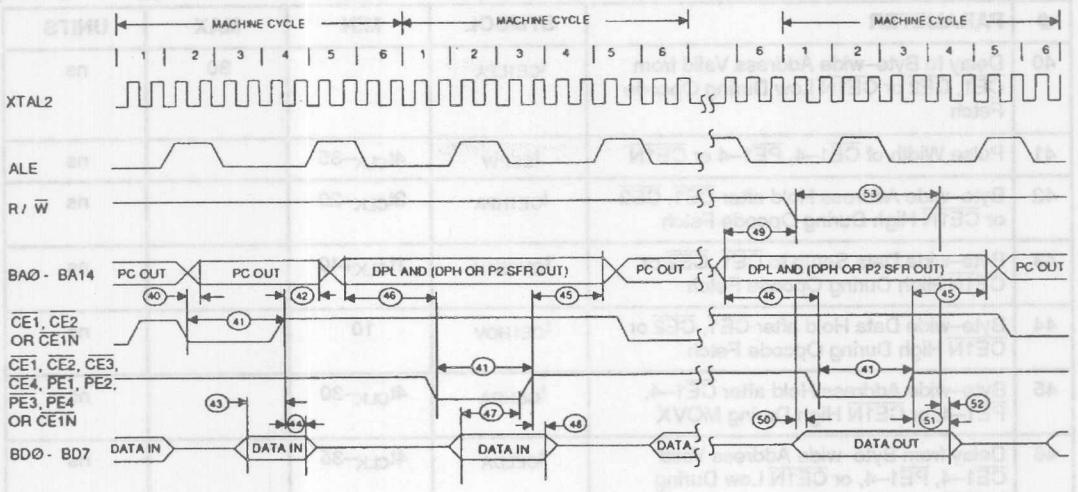


AC CHARACTERISTICS (cont'd)
PARALLEL PROGRAM LOAD TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Delay to Byte–wide Address Valid from $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ Low During Opcode Fetch	t_{CE1LPA}		30	ns
41	Pulse Width of $\overline{CE1}$–4, $\overline{PE1}$–4 or $\overline{CE1N}$	t_{CEPW}	$4t_{CLK}$–35		ns
42	Byte–wide Address Hold after $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HPA}	$2t_{CLK}$–20		ns
43	Byte–wide Data Setup to $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{OVCE1H}	$1t_{CLK}$+40		ns
44	Byte–wide Data Hold after $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HOV}	10		ns
45	Byte–wide Address Hold after $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX	t_{CEHDA}	$4t_{CLK}$–30		ns
46	Delay from Byte–wide Address Valid $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ Low During MOVX	t_{CELDA}	$4t_{CLK}$–35		ns
47	Byte–wide Data Setup to $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{DACEH}	$1t_{CLK}$+40		ns
48	Byte–wide Data Hold after $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{CEHDV}	10		ns
49	Byte–wide Address Valid to R/$\overline{W}$ Active During MOVX (write)	t_{AVRWL}	$3t_{CLK}$–35		ns
50	Delay from R/$\overline{W}$ Low to Valid Data Out During MOVX (write)	t_{RWLDV}	20		ns
51	Valid Data Out Hold Time from $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High	t_{CEHDV}	$1t_{CLK}$–15		ns
52	Valid Data Out Hold Time from R/$\overline{W}$ High	t_{RWHDV}	0		ns
53	Write Pulse Width (R/$\overline{W}$ Low Time)	t_{RWLPW}	$6t_{CLK}$–20		ns

BYTE–WIDE BUS TIMING



RPC AC CHARACTERISTICS – DBB READ

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
54	$\overline{CS}$, A_0 Setup to $\overline{RD}$	t_{AR}	0		ns
55	$\overline{CS}$, A_0 Hold After $\overline{RD}$	t_{RA}	0		ns
56	$\overline{RD}$ Pulse Width	t_{RR}	160		ns
57	$\overline{CS}$, A_0 to Data Out Delay	t_{AD}		130	ns
58	$\overline{RD}$ to Data Out Delay	t_{RD}	0	130	ns
59	$\overline{RD}$ to Data Float Delay	t_{RDZ}		85	ns

RPC AC CHARACTERISTICS – DBB WRITE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

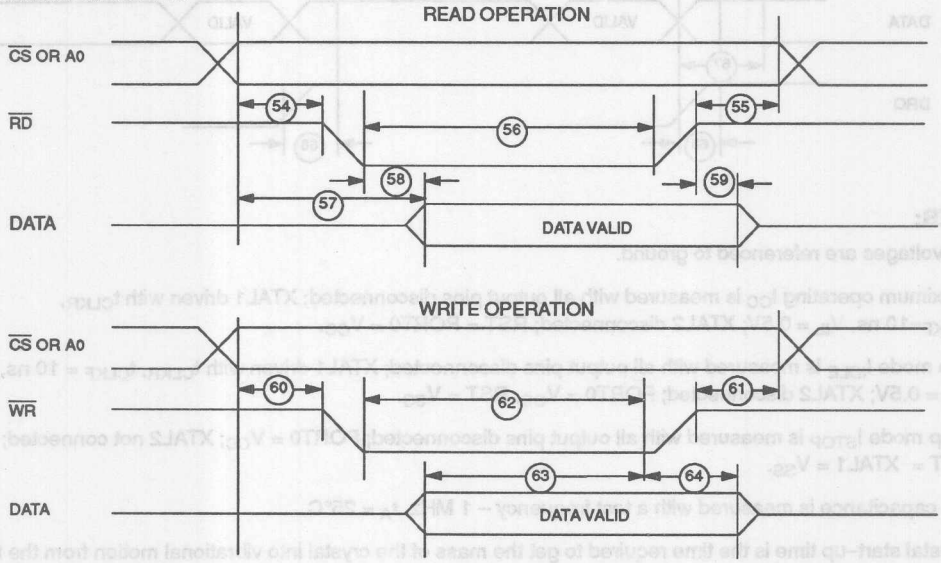
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
60	$\overline{CS}$, A_0 Setup to $\overline{WR}$	t_{AW}	0		ns
61A	$\overline{CS}$, Hold After $\overline{WR}$	t_{WA}	0		ns
61B	A_0, Hold After $\overline{WR}$	t_{WA}	20		ns
62	$\overline{WR}$ Pulse Width	t_{WW}	20		ns
63	Data Setup to $\overline{WR}$	t_{DW}	130		ns
64	Data Hold After $\overline{WR}$	t_{WD}	20		ns

AC CHARACTERISTICS – DMA

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
65	$\overline{DACK}$ to $\overline{WR}$ or $\overline{RD}$	t_{ACC}	0		ns
66	$\overline{RD}$ or $\overline{WR}$ to $\overline{DACK}$	t_{CAC}	0		ns
67	$\overline{DACK}$ to Data Valid	t_{ACD}	0	130	ns
68	$\overline{RD}$ or $\overline{WR}$ to DRQ Cleared	t_{CRQ}		110	ns

RPC TIMING MODE 16

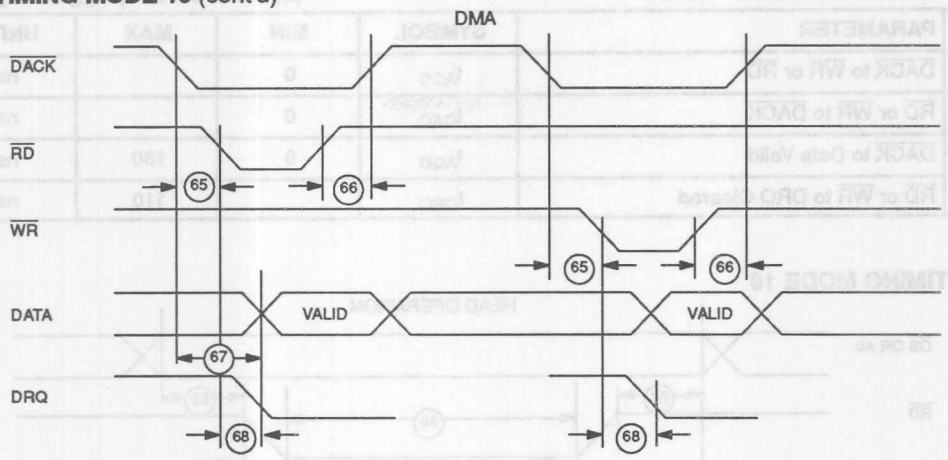


AC CHARACTERISTICS – $\overline{PROG}$

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
69	$\overline{PROG}$ Low to Active	t_{PRA}	48		CLKS
70	$\overline{PROG}$ High to Inactive	t_{PRI}	48		CLKS

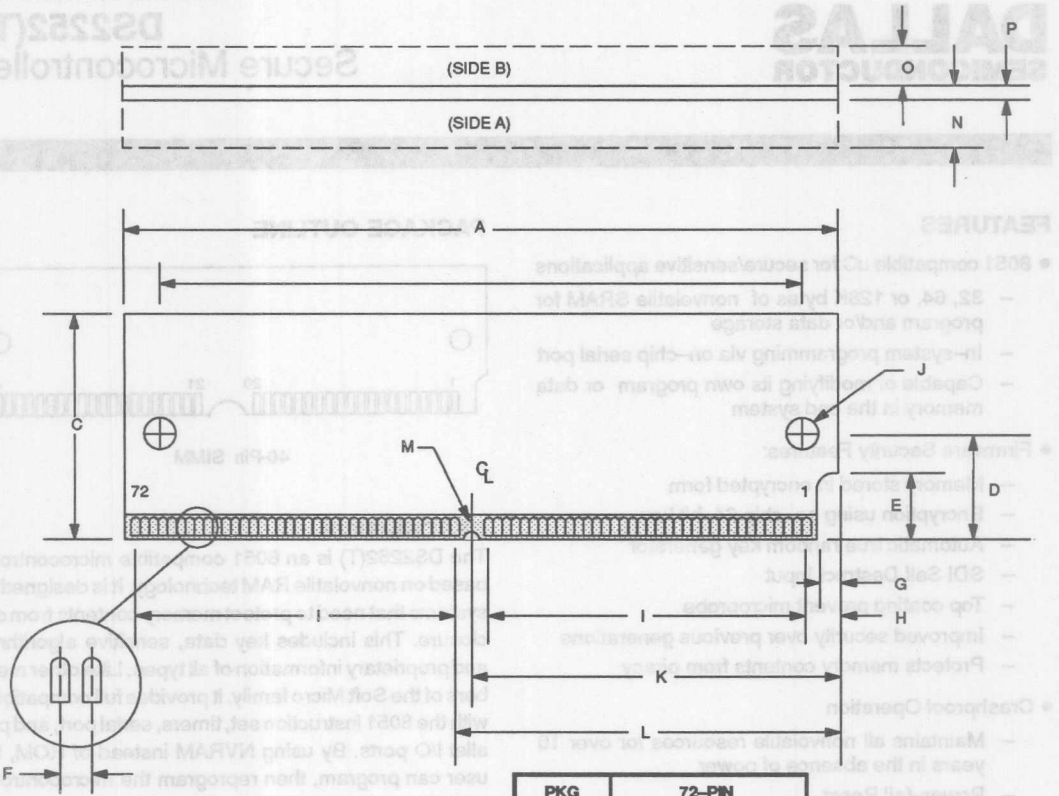
RPC TIMING MODE 16 (cont'd)



NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; RST = PORT0 = V_{CC}.
3. Idle mode I_{IDLE} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; PORT0 = V_{CC}, RST = V_{SS}.
4. Stop mode I_{STOP} is measured with all output pins disconnected; PORT0 = V_{CC}; XTAL2 not connected; RST = XTAL1 = V_{SS}.
5. Pin capacitance is measured with a test frequency – 1 MHz, t_A = 25°C.
6. Crystal start–up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on–chip oscillator. The user should check with the crystal vendor for a worst case specification on this time.

PACKAGE DRAWING



PKG	72-PIN	
DIM	MIN	MAX
A	4.245	4.255
B	3.979	3.989
C	0.995	1.005
D	0.395	0.405
E	0.245	0.255
F	0.050 BSC	
G	0.075	0.085
H	0.245	0.255
I	1.750 BSC	
J	0.120	0.130
K	2.120	2.130
L	2.245	2.255
M	0.057	0.067
N		0.275
O		0.145
P		0.054

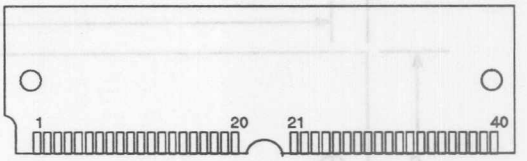
DALLAS SEMICONDUCTOR

DS2252(T)
Secure Microcontroller

FEATURES

- 8051 compatible uC for secure/sensitive applications
  - 32, 64, or 128K bytes of nonvolatile SRAM for program and/or data storage
  - In–system programming via on–chip serial port
  - Capable of modifying its own program or data memory in the end system
- Firmware Security Features:
  - Memory stored in encrypted form
  - Encryption using on–chip 64–bit key
  - Automatic true random key generator
  - SDI Self Destruct Input
  - Top coating prevent microprobe
  - Improved security over previous generations
  - Protects memory contents from piracy
- Crashproof Operation
  - Maintains all nonvolatile resources for over 10 years in the absence of power
  - Power–fail Reset
  - Early Warning Power–fail Interrupt
  - Watchdog Timer
  - Precision reference for power monitor
- Fully 8051 Compatible
  - 128 bytes scratchpad RAM
  - Two timer/counters
  - On–chip serial port
  - 32 parallel I/O port pins
- Optional permanently powered Real–time Clock (DS2252T)

PACKAGE OUTLINE



40-Pin SIMM

DESCRIPTION

The DS2252(T) is an 8051 compatible microcontroller based on nonvolatile RAM technology. It is designed for systems that need to protect memory contents from disclosure. This includes key data, sensitive algorithms, and proprietary information of all types. Like other members of the Soft Micro family, it provides full compatibility with the 8051 instruction set, timers, serial port, and parallel I/O ports. By using NVRAM instead of ROM, the user can program, then reprogram the microcontroller while in–system. This allows frequent changing of sensitive processes with minimal effort. The DS2252 provides an array of mechanisms to prevent an attacker from examining the memory. It is designed to resist all levels of threat including observation, analysis, and physical attack. As a result, a massive effort would be required to obtain any information about memory contents. Furthermore, the "Soft" nature of the DS2252 allows frequent modification of secure information. This minimizes that value of any information that is obtained.

Using a security system based on the DS5002, the DS2252 protects the memory contents from disclosure.

It loads program memory via its serial port and encrypts it in real–time prior to storing it in SRAM. Once encrypted, the RAM contents and the program flow are unintelligible. The real data exists only inside the processor chip after being decrypted. Any attempt to discover the on–chip data, encryption keys, etc., results in its destruction. Extensive use of nonvolatile lithium backed technology create a micro that retains data for over 10 years, but which can be erased instantly if tampered with. The DS2252 even interfaces directly to external tamper protection hardware.

The DS2252T provides a permanently powered Real–time Clock with interrupts for time stamp and date. It keeps time to one hundredth of second using its on–board 32 KHz crystal.

Like other Soft Micros in the family, the DS2252(T) provides crashproof operation in portable systems or systems with unreliable power. These features include the ability to save the operating state, Power–fail Reset, Power–fail Interrupt, and Watchdog Timer. All nonvolatile memory and resources are maintained for over 10 years at room temperature in the absence of power.

A user loads programs into the DS2252(T) via its on–chip Serial Bootstrap Loader. This function supervises the loading of software into NVRAM, validates it, then becomes transparent to the user. It also manages the loading of new encryption keys automatically. Software is stored in on–board CMOS SRAM. Using its internal Partitioning, the DS2252(T) can divide a common RAM into user selectable program and data segments. This Partition can be selected at program loading time, but can be modified anytime later. The micro will decode memory access to the SRAM, access memory via its Byte–wide bus and write–protect the memory portion designated as program (ROM).

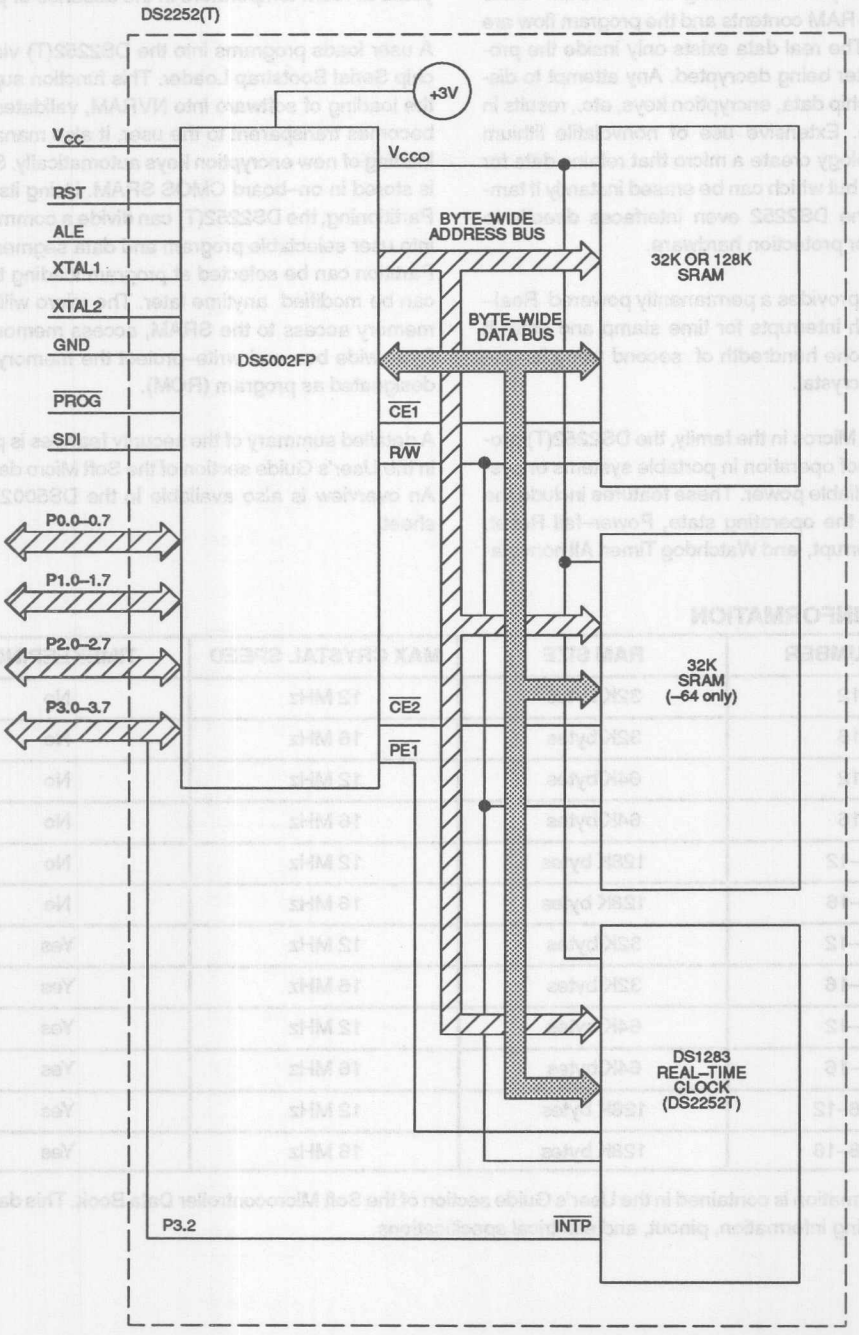
A detailed summary of the security features is provided in the User's Guide section of the Soft Micro data book. An overview is also available in the DS5002FP data sheet.

ORDERING INFORMATION

PART NUMBER	RAM SIZE	MAX CRYSTAL SPEED	TIMEKEEPING?
DS2252–32–12	32K bytes	12 MHz	No
DS2252–32–16	32K bytes	16 MHz	No
DS2252–64–12	64K bytes	12 MHz	No
DS2252–64–16	64K bytes	16 MHz	No
DS2252–128–12	128K bytes	12 MHz	No
DS2252–128–16	128K bytes	16 MHz	No
DS2252T–32–12	32K bytes	12 MHz	Yes
DS2252T–32–16	32K bytes	16 MHz	Yes
DS2252T–64–12	64K bytes	12 MHz	Yes
DS2252T–64–16	64K bytes	16 MHz	Yes
DS2252T–128–12	128K bytes	12 MHz	Yes
DS2252T–128–16	128K bytes	16 MHz	Yes

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pinout, and electrical specifications.

DS2252(T) BLOCK DIAGRAM Figure 1



PIN ASSIGNMENT

1	P1.0	11	P1.5	21	P3.1 TXD	31	P3.6 $\overline{WR}$
2	V_{CC}	12	P0.4	22	ALE	32	P2.4
3	P1.1	13	P1.6	23	P3.2 $\overline{INT0}$	33	P3.7 $\overline{RD}$
4	P0.0	14	P0.5	24	$\overline{PROG}$	34	P2.3
5	P1.2	15	P1.7	25	P3.3 $\overline{INT1}$	35	XTAL2
6	P0.1	16	P0.6	26	P2.7	36	P2.2
7	P1.3	17	RST	27	P3.4 T0	37	XTAL1
8	P0.2	18	P0.7	28	P2.6	38	P2.1
9	P1.4	19	P3.0 RXD	29	P3.5 T1	39	GND
10	P0.3	20	SDI	30	P2.5	40	P2.0

PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
4, 6, 8, 10, 12, 14, 16, 18	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull–ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
1, 3, 5, 7, 9, 11, 13, 15	P1.0–P1.7 General purpose I/O Port 1.
40, 38, 36, 34, 32, 30, 28, 26	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.
19	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
21	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
23	P3.2 $\overline{INT0}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
25	P3.3 $\overline{INT1}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
27	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
29	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
31	P3.6 $\overline{WR}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.

PIN NUMBER	DESCRIPTION
33	P3.7 $\overline{RD}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
17	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally, can be left unconnected if not used. An RC power–on reset circuit is not needed and is NOT recommended.
22	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch.
35, 37	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
39	GND Logic ground.
2	V_{CC} +5V
24	$\overline{PROG}$ Invokes the Bootstrap loader on a falling edge. This signal should be debounced so that only one edge is detected. If connected to ground, the micro will enter Bootstrap loading on power up. This signal is pulled up internally.
20	SDI Self Destruct Input. A logic 1 applied to this input causes a hardware unlock. This involves the destruction of Encryption Keys, Vector RAM, and the momentary removal of power from V_{CCO}. This pin should be grounded if not used. To activate, it should be taken to a logic 1 or +3V.

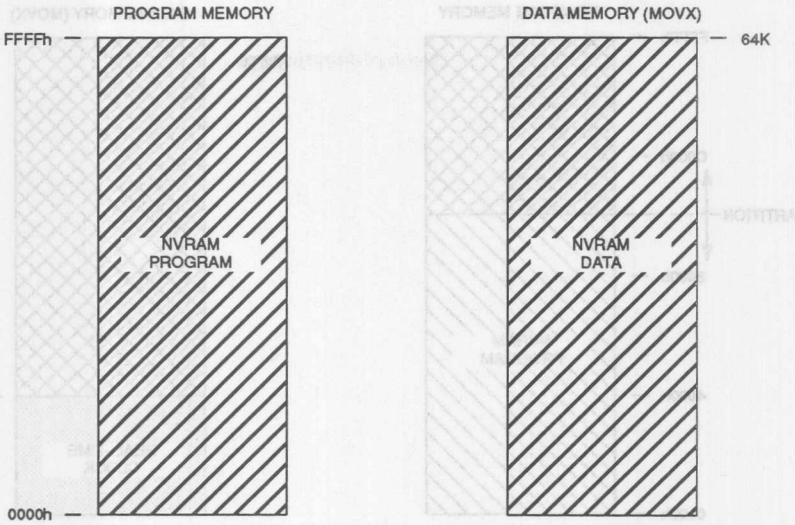
INSTRUCTION SET

The DS2252(T) executes an instruction set that is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages such as assemblers and compilers that have been written for the 8051 are compatible with the DS2252(T). A complete description of the instruction set and operation are provided in the User's Guide section of the Soft Microcontroller Data Book.

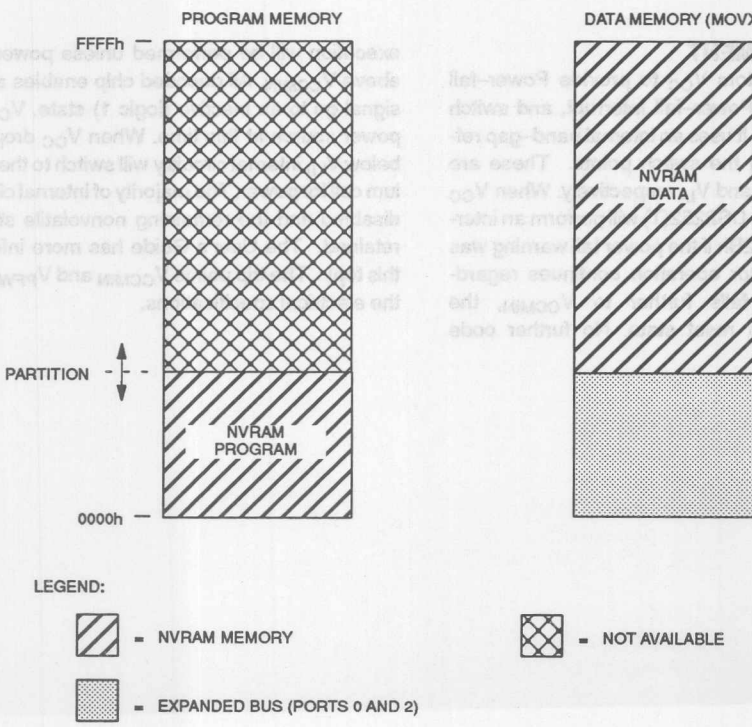
MEMORY ORGANIZATION

Figure 2 illustrates the memory map accessed by the DS2252(T). The entire 64K of program and 64K of data are available to the Byte–wide bus. This preserves the I/O ports for application use. An alternate configuration allows dynamic Partitioning of a 64K space as shown in Figure 3. Any data area not mapped into the NVRAM is reached via the Expanded bus on Ports 0 and 2. Off–board program memory is not available for security reasons. Selecting PES=1 provides access to the Real–time Clock on the DS2252T as shown in Figure 4. These selections are made using Special Function Registers. The memory map and its controls are covered in detail in the User's Guide section of the Soft Microcontroller Data Book.

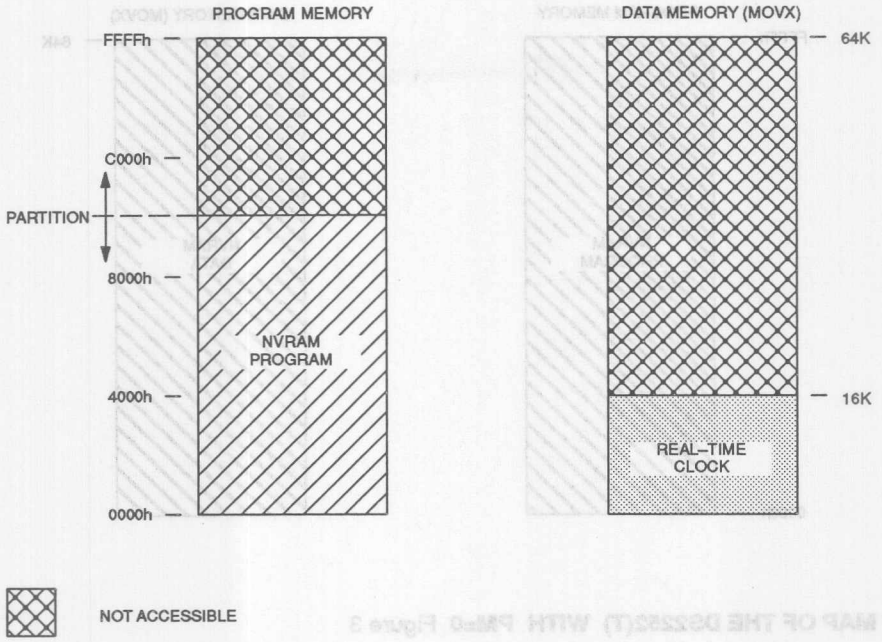
MEMORY MAP OF THE DS2252(T) WITH PM=1 Figure 2



MEMORY MAP OF THE DS2252(T) WITH PM=0 Figure 3



MEMORY MAP OF THE DS2252(T) WITH PES=1 Figure 4



POWER MANAGEMENT

The DS2252(T) monitors V_{CC} to provide Power–fail Reset, early warning Power–fail Interrupt, and switch over to lithium backup. It uses an internal band–gap reference in determining the switch points. These are called V_{PFW}, V_{CCMIN}, and V_{LI} respectively. When V_{CC} drops below V_{PFW}, the DS2252(T) will perform an interrupt vector to location 2Bh if the power fail warning was enabled. Full processor operation continues regardless. When power falls further to V_{CCMIN}, the DS2252(T) invokes a reset state. No further code execution will be performed unless power rises back above V_{CCMIN}. All decoded chip enables and the R/$\overline{W}$ signal go to an inactive (logic 1) state. V_{CC} is still the power source at this time. When V_{CC} drops further to below V_{LI}, internal circuitry will switch to the built–in lithium cell for power. The majority of internal circuits will be disabled and the remaining nonvolatile states will be retained. The User's Guide has more information on this topic. The trip points V_{CCMIN} and V_{PFW} are listed in the electrical specifications.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A=0°C to 70°C; V_{CC}=5V ± 10%)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	–0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC}+0.3	V	1
Input High Voltage (RST, XTAL1, $\overline{PROG}$)	V_{IH2}	3.5		V_{CC}+0.3	V	1
Output Low Voltage @ I_{OL}=1.6 mA (Ports 1, 2, 3)	V_{OL1}		0.15	0.45	V	1
Output Low Voltage @ I_{OL}=3.2 mA (Ports 0, ALE)	V_{OL2}		0.15	0.45	V	1
Output High Voltage @ I_{OH}=–80 µA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @ I_{OH}=–400 µA (Ports 0, ALE)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN}=0.45V (Ports 1, 2, 3)	I_{IL}			–50	µA	
Transition Current; 1 to 0 V_{IN}=2.0V (Ports 1, 2, 3)	I_{TL}			–500	µA	
Input Leakage Current 0.45<V_{IN}<V_{CC} (Port 0)	I_{IL}			±10	µA	
RST Pulldown Resistor	R_{RE}	40		150	KΩ	
Power Fail Warning Voltage	V_{PRW}	4.25	4.37	4.50	V	1
Minimum Operating Voltage	V_{CCMIN}	4.00	4.12	4.25	V	1
Operating Current	I_{CC}			45	mA	4
Idle Mode Current	I_{IDLE}			7.0	mA	5
Stop Mode current	I_{STOP}			80	µA	6
Pin Capacitance	C_{IN}			10	pF	7
Reset Trip Point in Stop Mode w/BAT=3.0V w/BAT=3.3V		 4.0 4.4		 4.25 4.65	V	1
SDI Input Low Voltage	V_{ILS}			0.4	V	1
SDI Input High Voltage	V_{IHS}	2.0		V_{CC}	V	1, 2
SDI Input High Voltage	V_{IHS}	2.0		3.5	V	1, 2
SDI Pull–Down Resistor	R_{SDI}	25		60	KΩ	

AC CHARACTERISTICS

(t_A = 0°C to70°C; V_{CC}=0V to 5V)

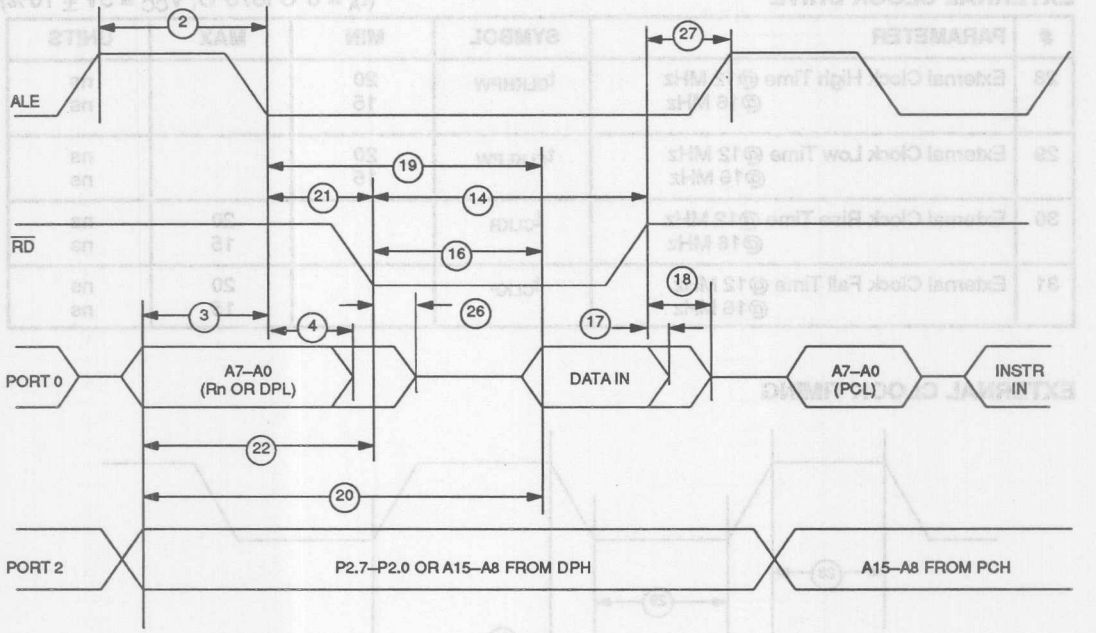
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
SDI Pulse Reject	t_{SPR}			2	µs	3
SDI Pulse Accept	t_{SPA}	10			µs	3

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

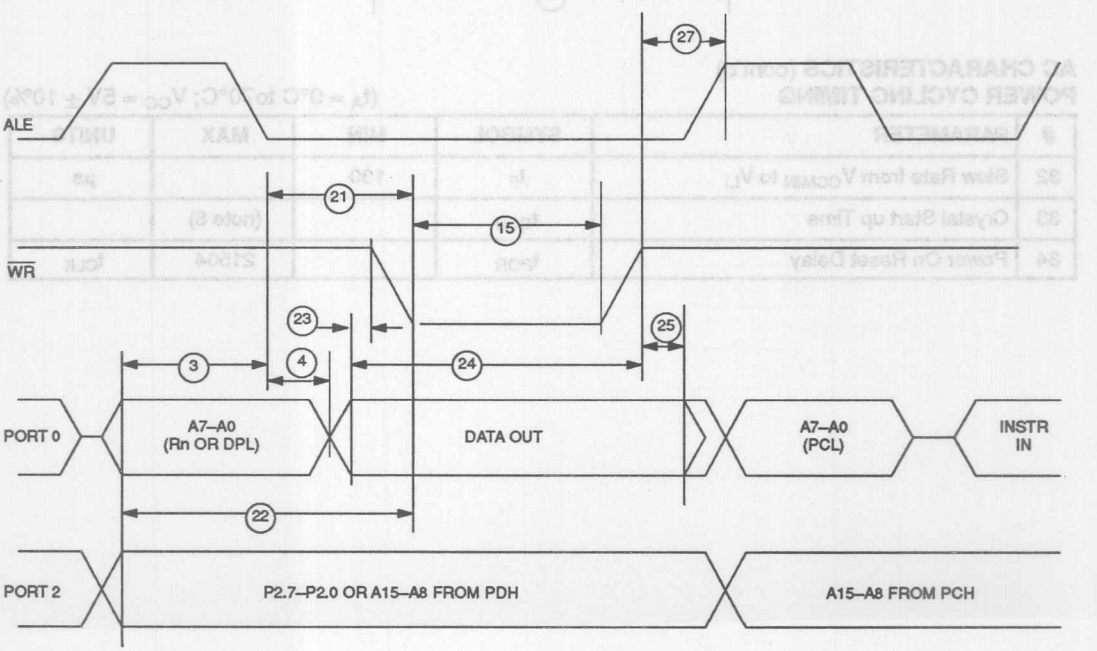
(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2t_{CLK}$-40		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK}-40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK}-35		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$-100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$-100		ns
16	$\overline{RD}$ Low to Valid Data In @12 MHz @16 MHz	t_{RDLDV}		$5t_{CLK}$-165 $5t_{CLK}$-105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$-70	ns
19	ALE Low to Valid Data In @12 MHz @16 MHz	t_{ALLVD}		$8t_{CLK}$-150 $8t_{CLK}$-90	ns ns
20	Valid Addr. to Valid Data In @12 MHz @16 MHz	t_{AVDV}		$9t_{CLK}$-165 $9t_{CLK}$-105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$-50	$3t_{CLK}$+50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$-130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK}-60		ns
24	Data Valid to $\overline{WR}$ High @12 MHz @16 MHz	t_{DVWRH}	$7t_{CLK}$-150 $7t_{CLK}$-90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK}-50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK}-40	t_{CLK}+50	ns

EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE

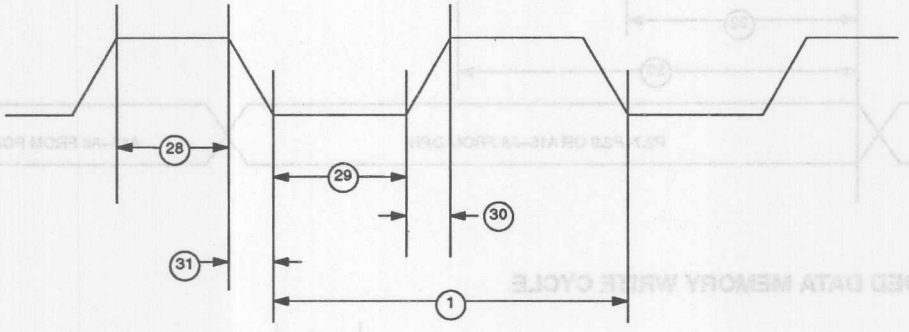


AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @12 MHz @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @12 MHz @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @12 MHz @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @12 MHz @16 MHz	t_{CLKF}		20 15	ns ns

EXTERNAL CLOCK TIMING

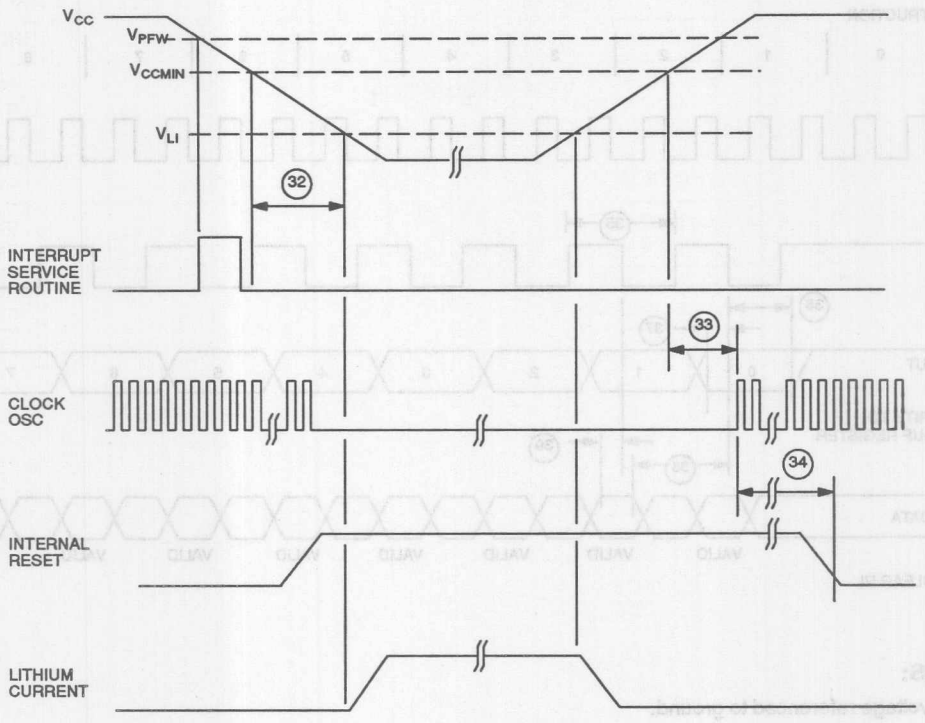


AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCMIN} to V_{LI}	t_F	130		µs
33	Crystal Start up Time	t_{CSU}		(note 8)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

POWER CYCLE TIMING

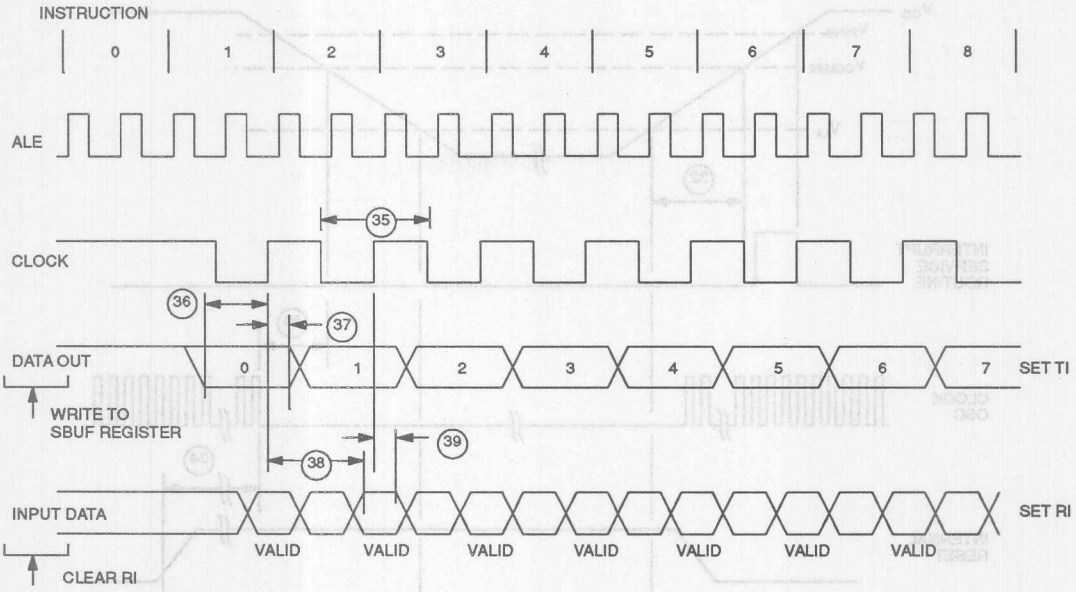


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Clock Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$-133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$-117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$-133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

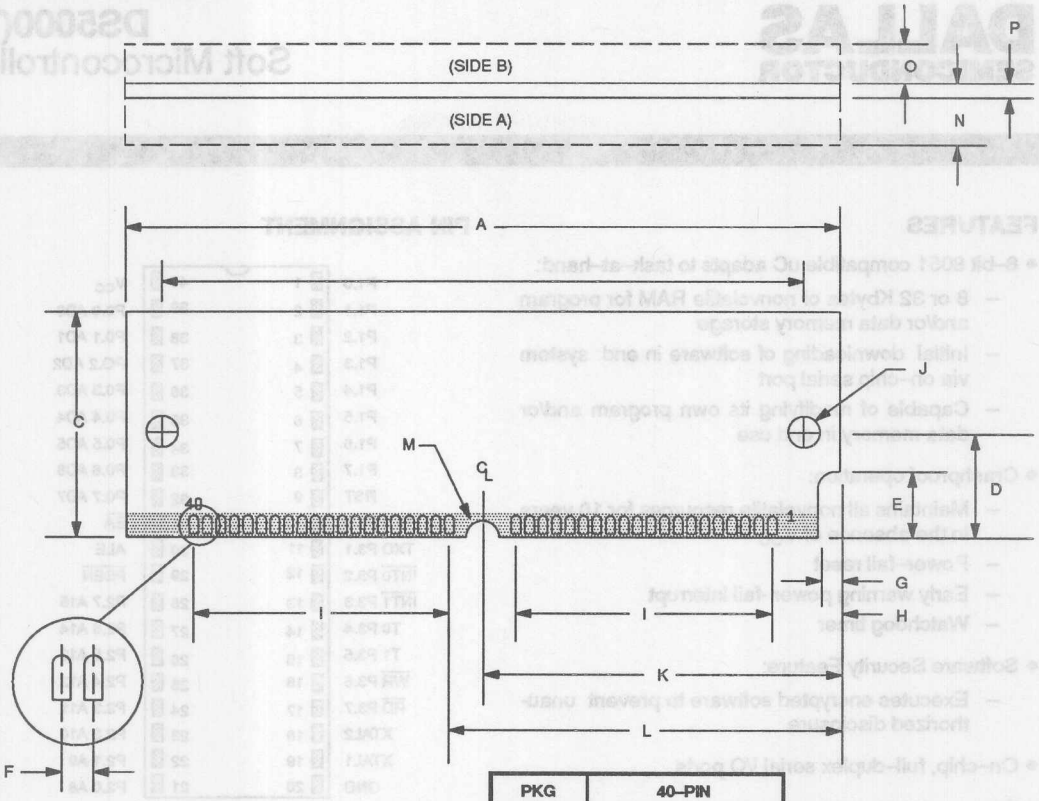
SERIAL PORT TIMING – MODE 0



NOTES:

1. All voltage referenced to ground.
2. SDI should be taken to a logic high when V_{CC}=+5V, and to approximately 3V when V_{CC}<3V.
3. SDI is deglitched to prevent accidental destruction. The pulse must be longer than t_{SPR} to pass the deglitcher, but SDI is not guaranteed unless it is longer than t_{SPA}.
4. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; RST = PORT0 = V_{CC}.
5. Idle mode I_{IDLE} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; PORT0 = V_{CC}, RST = V_{SS}.
6. Stop mode I_{STOP} is measured with all output pins disconnected; PORT0 = V_{CC}; XTAL2 not connected; RST = XTAL1 = V_{SS}.
7. Pin capacitance is measured with a test frequency – 1 MHz, t_A = 25°C.
8. Crystal start–up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on–chip oscillator. The user should check with the crystal vendor for a worst case specification on this time.

PACKAGE DRAWING



PKG	40–PIN	
DIM	MIN	MAX
A	2.645	2.655
B	2.379	2.389
C	0.995	1.005
D	0.395	0.405
E	0.245	0.255
F	0.050 BSC	
G	0.075	0.085
H	0.245	0.255
I	0.950 BSC	
J	0.120	0.130
K	1.320	1.330
L	1.445	1.455
M	0.057	0.067
N		0.300
O		0.165
P		0.054

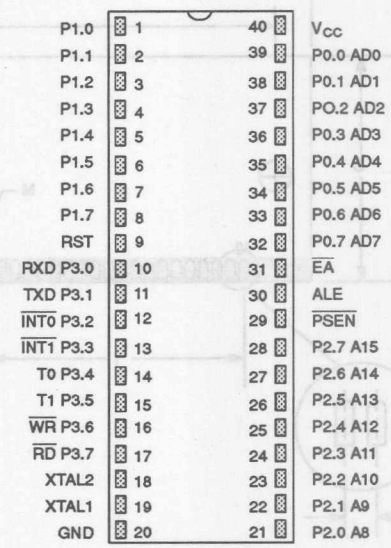
DALLAS SEMICONDUCTOR

DS5000(T)
Soft Microcontroller

FEATURES

- 8–bit 8051 compatible uC adapts to task–at–hand:
  - 8 or 32 Kbytes of nonvolatile RAM for program and/or data memory storage
  - Initial downloading of software in end system via on–chip serial port
  - Capable of modifying its own program and/or data memory in end use
- Crashproof operation:
  - Maintains all nonvolatile resources for 10 years in the absence of V_{CC}
  - Power–fail reset
  - Early warning power–fail interrupt
  - Watchdog timer
- Software Security Feature:
  - Executes encrypted software to prevent unauthorized disclosure
- On–chip, full–duplex serial I/O ports
- Two on–chip timer/event counters
- 32 parallel I/O lines
- Compatible with industry standard 8051 instruction set and pinout
- Optional Permanently Powered Real–Time Clock (DS5000T)

PIN ASSIGNMENT



40-Pin Encapsulated Package

DESCRIPTION

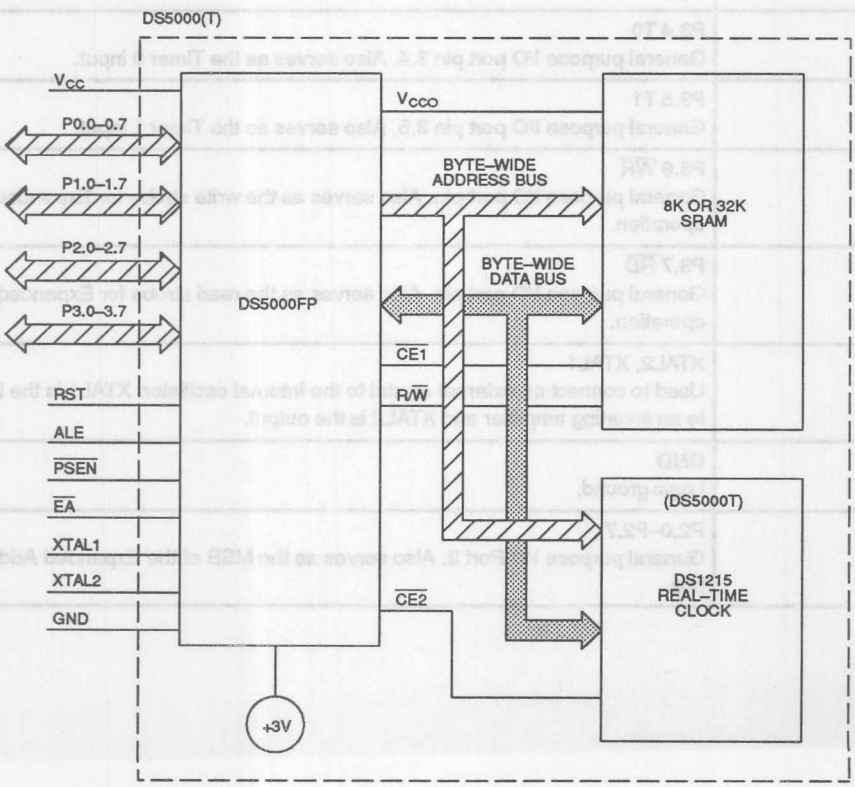
The DS5000(T) Soft Microcontroller is a fully 8051 compatible 8–bit CMOS microcontroller that offers "softness" in all aspects of its application. This is accomplished through the comprehensive use of nonvolatile technology to preserve all information in the absence of system V_{CC}. The internal program/data memory space is implemented using either 8K or 32K bytes of nonvolatile CMOS SRAM. Furthermore, internal data registers and key configuration registers are also nonvolatile. An optional real–time clock gives permanently powered timekeeping. The clock keeps time to a hundredth of a second using an on–board crystal.

ORDERING INFORMATION

PART NUMBER	RAM SIZE	MAX CRYSTAL SPEED	TIMEKEEPING?
DS5000–8–8	8K bytes	8 MHz	No
DS5000–8–12	8K bytes	12 MHz	No
DS5000–8–16	8K bytes	16 MHz	No
DS5000–32–8	32K bytes	8 MHz	No
DS5000–32–12	32K bytes	12 MHz	No
DS5000–32–16	32K bytes	16 MHz	No
DS5000T–8–8	8K bytes	8 MHz	Yes
DS5000T–8–12	8K bytes	12 MHz	Yes
DS5000T–8–16	8K bytes	16 MHz	Yes
DS5000T–32–8	32K bytes	8 MHz	Yes
DS5000T–32–12	32K bytes	12 MHz	Yes
DS5000T–32–16	32K bytes	16 MHz	Yes

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pinout, and electrical specification.

DS5000(T) BLOCK DIAGRAM Figure 1



PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
1–8	P1.0 – P1.7 General purpose I/O Port 1
9	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally so this pin can be left unconnected if not used.
10	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should not be connected directly to a PC COM port.
11	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should not be connected directly to a PC COM port.
12	P3.2 $\overline{\text{INT0}}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
13	P3.3 $\overline{\text{INT1}}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
14	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
15	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
16	P3.6 $\overline{\text{WR}}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
17	P3.7 $\overline{\text{RD}}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
18, 19	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
20	GND Logic ground.
21–28	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.

PIN NUMBER	DESCRIPTION
29	$\overline{PSEN}$ Program Store Enable. This active low signal is used to enable an external program memory when using the Expanded bus. It is normally an output and should be unconnected if not used. $\overline{PSEN}$ also is used to invoke the Bootstrap Loader. At this time, $\overline{PSEN}$ will be pulled down externally. This should only be done once the DS5000 is already in a reset state. The device that pulls down should be open drain since it must not interfere with $\overline{PSEN}$ under normal operation.
30	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch. When using a parallel programmer, this pin also assumes the $\overline{PROG}$ function for programming pulses.
31	$\overline{EA}$ External Access. This pin forces the DS5000 to behave like an 8031. No internal memory (or clock) will be available when this pin is at a logic low. Since this pin is pulled down internally, it should be connected to +5V to use NVRAM. In a parallel programmer, this pin also serves as V_{PP} for super voltage pulses.
32–39	P0.7–P0.0 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull–ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
40	V_{CC} +5 volts.

INSTRUCTION SET

The DS5000 executes an instruction set which is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages which have been written for the 8051 are compatible with the DS5000, including cross–assemblers, high–level language compilers, and debugging tools.

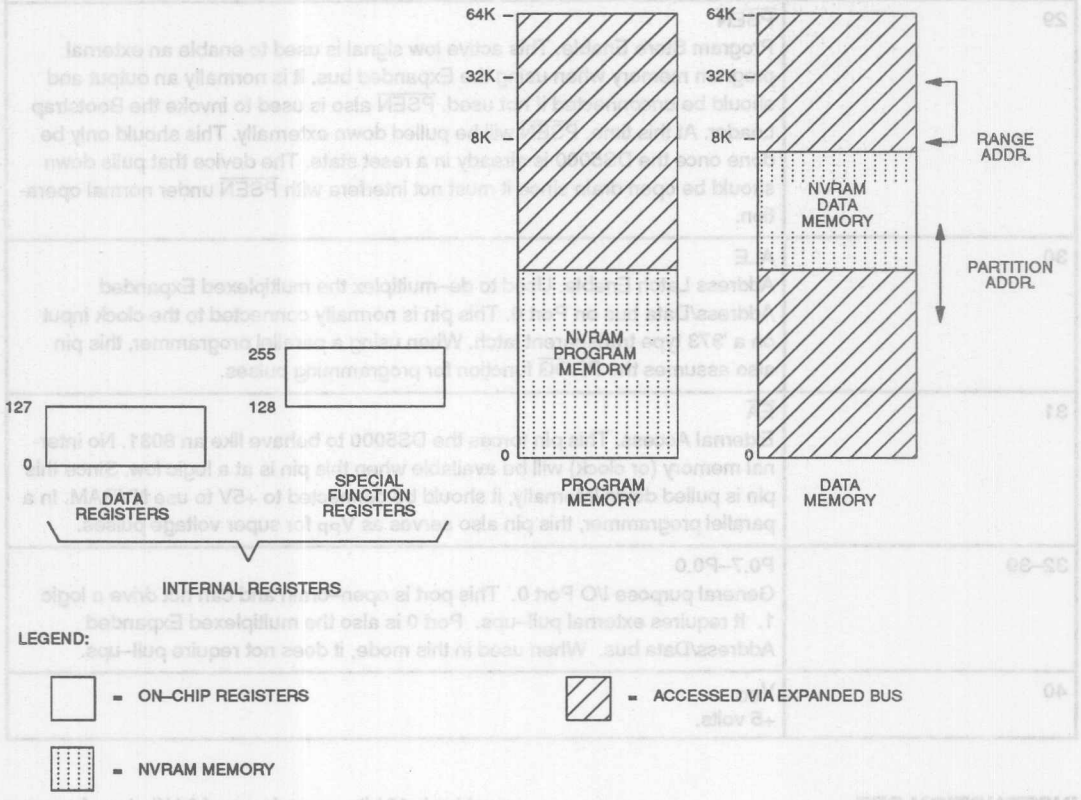
A complete description for the DS5000 instruction set is available in the User's Guide section of the Soft Microcontroller Data Book.

MEMORY ORGANIZATION

Figure 2 illustrates the address spaces which are accessed by the DS5000. As illustrated in the figure, separate address spaces exist for program and data memory. Since the basic addressing capability of the machine is 16 bits, a maximum of 64 Kbytes of program memory and 64 Kbytes of data memory can be accessed by the DS5000 CPU. The 8K or 32K byte RAM area inside of the DS5000 can be used to contain both program and data memory.

The Real–time Clock (RTC) in the DS5000T is reached in the memory map by setting a SFR bit. The MCON.2 bit (ECE2) is used to select an alternate data memory map. While ECE2=1, all MOVXs will be routed to this alternate memory map. The real–time clock is a serial device that resides in this area. A full description of the RTC access and example software is given in the User's Guide section of the Soft Microcontroller Data Book. If the ECE2 bit is set on a DS5000 without a timekeeper, the MOVXs will simply go to a nonexistent memory. Software execution would not be affected otherwise.

DS5000 LOGICAL ADDRESS SPACES Figure 2



PROGRAM LOADING

The Program Load Modes allow initialization of the NVRAM Program/Data Memory. This initialization may be performed in one of two ways:

1. Serial Program Loading which is capable of performing Bootstrap Loading of the DS5000(T). This feature allows the loading of the application program to be delayed until the DS5000(T) is installed in the end system.
2. Parallel Program Load cycles which perform the initial loading from parallel address/data information presented on the I/O port pins. this mode is timing–set compatible with the 8751H microcontroller programming mode.

The DS5000 is placed in its Program Load configuration by simultaneously applying a logic 1 to the RST pin and forcing the $\overline{\text{PSEN}}$ line to a logic 0 level. Immediately following this action, the DS5000 will look for a parallel Program Load pulse, or a serial ASCII carriage return (0DH) character received at 9600, 2400, 1200, or 300 bps over the serial port.

The hardware configurations used to select these modes of operation are illustrated in Figure 3.

PROGRAM LOADING CONFIGURATIONS Figure 3

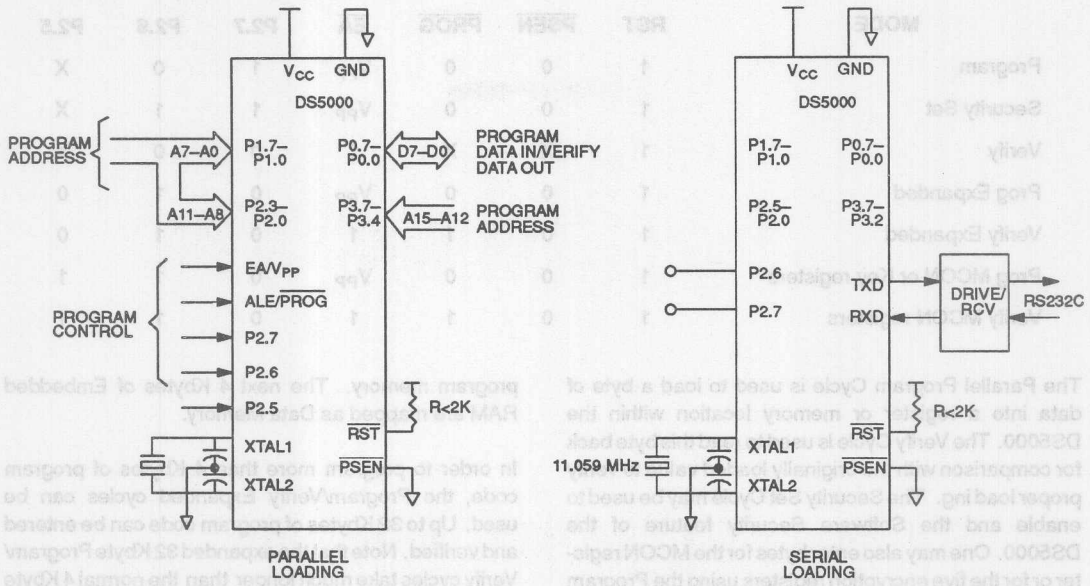


Table 1 summarizes the selection of the available Parallel Program Load cycles. The timing associated with these cycles is illustrated in the electrical specs.

SERIAL BOOTSTRAP LOADER

The Serial Program Load Mode is the easiest, fastest, most reliable, and most complete method of initially loading application software into the DS5000 nonvolatile RAM. Communication can be performed over a standard asynchronous serial communications port. A typical application would use a simple RS232C serial interface to program the DS5000 as a final production procedure. The hardware configuration which is required for the Serial Program Load mode is illustrated in Figure 3. Port pins 2.7 and 2.6 must be either open or pulled high to avoid placing the DS5000 in a parallel load cycle. Although an 11.0592 MHz crystal is shown in Figure 3, a variety of crystal frequencies and loader baud rates are supported, shown in Table 2. The serial loader is designed to operate across a three–wire interface from a standard UART. The receive, transmit, and ground wires are all that are necessary to establish communication with the DS5000.

The Serial Bootstrap Loader implements an easy–to–use command line interface which allows an application program in an Intel hex representation to be loaded into and read back from the device. Intel hex is the typical format which existing 8051 cross–assemblers output. The serial loader responds to single character commands which are summarized below:

COMMAND	FUNCTION
C	Return CRC–16 checksum of embedded RAM
D	Dump Intel Hex File
F	Fill embedded RAM block with constant
K	Load 40–bit Encryption Key
L	Load Intel Hex File
R	Read MCON register
T	Trace (Echo) incoming Intel Hex data
U	Clear Security Lock
V	Verify Embedded RAM with incoming Intel Hex
W	Write MCON register
Z	Set Security Lock
P	Put a value to a port.
G	Get a value from a port.

MODE	RST	PSEN	PROG	EA	P2.7	P2.6	P5.0
Program	1	0	0	V_{PP}	1	0	X
Security Set	1	0	0	V_{PP}	1	1	X
Verify	1	X	X	1	0	0	X
Prog Expanded	1	0	0	V_{PP}	0	1	0
Verify Expanded	1	0	1	1	0	1	0
Prog MCON or Key registers	1	0	0	V_{PP}	0	1	1
Verify MCON registers	1	0	1	1	0	1	1

The Parallel Program Cycle is used to load a byte of data into a register or memory location within the DS5000. The Verify Cycle is used to read this byte back for comparison with the originally loaded value to verify proper load ing. The Security Set Cycle may be used to enable and the Software Security feature of the DS5000. One may also enter bytes for the MCON register or for the five encryption registers using the Program MCON cycle. When using this cycle, the absolute register address must be presented at Ports 1 and 2 as in the normal program cycle (Port 2 should be 00H). The MCON contents can likewise be verified using the Verify MCON cycle.

When the DS5000 first detects a Parallel Program Strobe pulse or a Security Set Strobe pulse while in the Program Load Mode following a Power On Reset, the internal hardware of the DS5000 is initialized so that an existing 4 Kbyte program can be programmed into a DS5000 with little or no modification. This initialization automatically sets the Range Address for 8 Kbytes and maps the lowest 4 Kbyte bank of Embedded RAM as program memory. The next 4 Kbytes of Embedded RAM are mapped as Data Memory.

In order to program more than 4 Kbytes of program code, the Program/Verify Expanded cycles can be used. Up to 32 Kbytes of program code can be entered and verified. Note that the expanded 32 Kbyte Program/Verify cycles take much longer than the normal 4 Kbyte Program/Verify cycles.

A typical parallel loading session would follow this procedure. First, set the contents of the MCON register with the correct range and partition only if using expanded programming cycles. Next, the encryption registers can be loaded to enable encryption of the program/data memory (not required). Then, program the DS5000 using either normal or expanded program cycles and check the memory contents using Verify cycles. The last operation would be to turn on the security lock feature by either a Security Set cycle or by explicitly writing to the MCON register and setting MCON.0 to a 1.

SERIAL LOADER BAUD RATES FOR DIFFERENT CRYSTAL FREQUENCIES Table 2

CRYSTAL FREQ (MHz)	BAUD RATE					
	300	1200	2400	9600	19200	57600
14.7456		Y	Y	Y	Y	
11.0592	Y	Y	Y	Y	Y	Y
9.21600	Y	Y	Y	Y		
7.37280	Y	Y	Y	Y		
5.52960	Y	Y	Y	Y		
1.84320	Y	Y	Y	Y		

ADDITIONAL INFORMATION

A complete description for all operational aspects of the DS5000, is provided in the User's Guide section of the Soft Microcontroller Data Book.

DEVELOPMENT SUPPORT

Dallas Semiconductor offers a kit package for developing and testing user code. The DS5000TK Evaluation Kit allows the user to download Intel hex formatted code directly to the DS5000 from a PC–XT/AT or compatible computer. The kit consists of a DS5000T–32–12, an interface pod, demo software, and an RS232 connector that attaches to the COM1 or COM2 serial port of a PC. See the User's Guide for further details.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

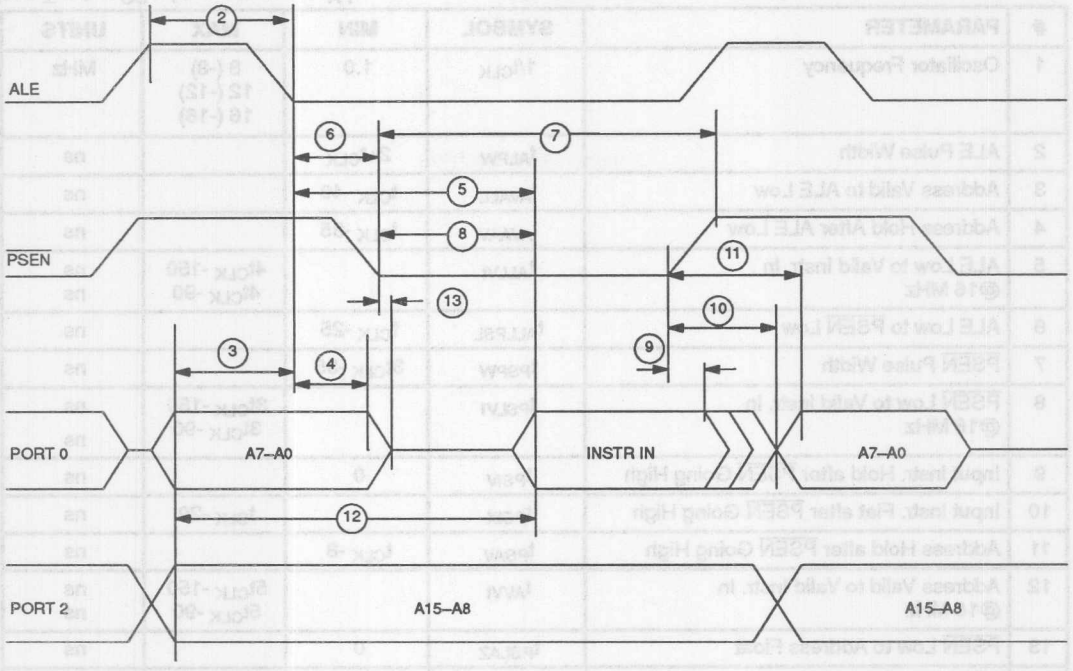
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	-0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC} +0.3	V	1
Input High Voltage RST, XTAL2	V_{IH2}	3.5		V_{CC} +0.3	V	1
Output Low Voltage @ I_{OL}=1.6mA (Ports 1, 2, 3)	V_{OL1}		.15	0.45	V	
Output Low Voltage @ I_{OL}=3.2mA (Ports 0, ALE, $\overline{PSEN}$)	V_{OL2}		.15	0.45	V	1
Output High Voltage @ I_{OH}=-80µA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @ I_{OH}=-400 µA (Ports 0, ALE, $\overline{PSEN}$)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN} = 0.45V (Ports 1, 2, 3)	I_{IL}			-50	µA	
Transition Current; 1 to 0 V_{IN} = 2.0V (Ports 1, 2, 3)	I_{TL}			-500	µA	
Input Leakage Current 0.45 < V_{IN} < V_{CC} (Port 0)	I_L			±10	µA	
RST, $\overline{EA}$ Pulldown Resistor	R_{RE}	40		125	KΩ	
Stop Mode Current	I_{SM}			80	µA	4
Power Fail Warning Voltage	V_{PFW}	4.15	4.6	4.75	V	1
Minimum Operating Voltage	V_{CCmin}	4.05	4.5	4.65	V	1
Programming Supply Voltage (Parallel Program Mode)	V_{PP}	12.5		13	V	1
Program Supply Current	I_{PP}		15	20	mA	
Operating Current DS5000–8K@8 MHz DS5000–32K @ 12 MHz DS5000T–32–16 @ 16 MHz	I_{CC}		25.2 35.7 45.6	43 48 54	mA	2
Idle Mode Current @ 12 MHz	I_{CC}		4.5	6.2	mA	3

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

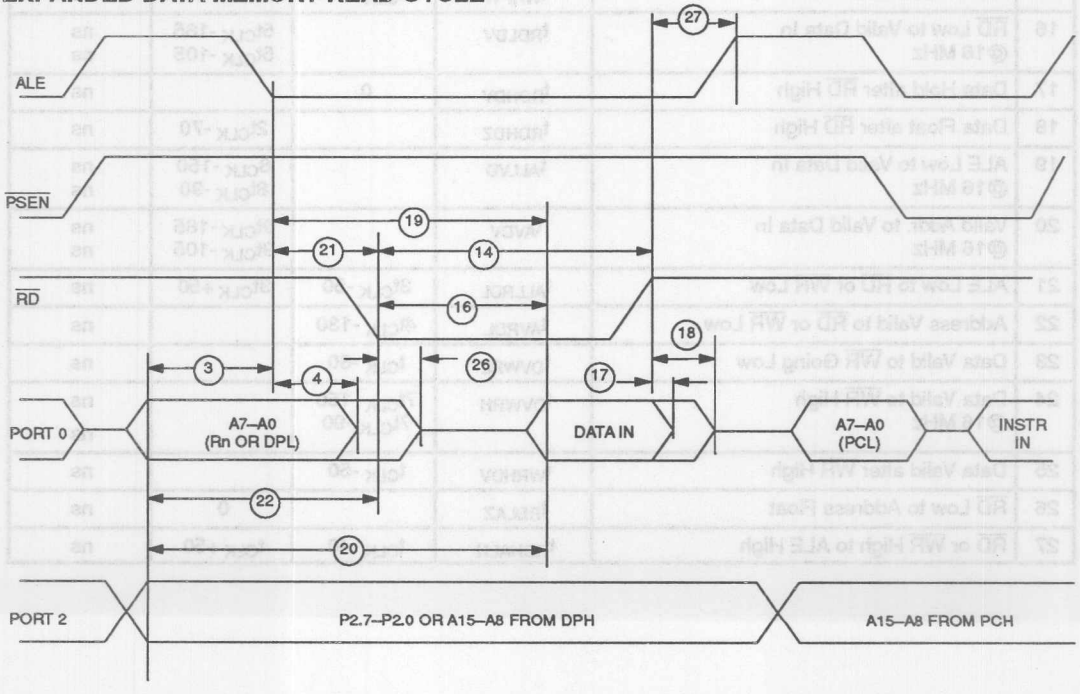
(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	8 (-8) 12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2*t_{CLK}$		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK} -40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK} -35		ns
5	ALE Low to Valid Instr. In @16 MHz	t_{ALLVI}		$4t_{CLK}$ -150 $4t_{CLK}$ -90	ns ns
6	ALE Low to $\overline{PSEN}$ Low	t_{ALLPSL}	t_{CLK} -25		ns
7	$\overline{PSEN}$ Pulse Width	t_{PSPW}	$3t_{CLK}$ -35		ns
8	$\overline{PSEN}$ Low to Valid Instr. In @16 MHz	t_{PSLVI}		$3t_{CLK}$ -150 $3t_{CLK}$ -90	ns ns
9	Input Instr. Hold after $\overline{PSEN}$ Going High	t_{PSIV}	0		ns
10	Input Instr. Flat after $\overline{PSEN}$ Going High	t_{PSIX}		t_{CLK} -20	ns
11	Address Hold after $\overline{PSEN}$ Going High	t_{PSAV}	t_{CLK} -8		ns
12	Address Valid to Valid Instr. In @16 MHz	t_{AVVI}		$5t_{CLK}$ -150 $5t_{CLK}$ -90	ns ns
13	$\overline{PSEN}$ Low to Address Float	t_{PSLAZ}	0		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$ -100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$ -100		ns
16	$\overline{RD}$ Low to Valid Data In @16 MHz	t_{RDLDV}		$5t_{CLK}$ -165 $5t_{CLK}$ -105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$ -70	ns
19	ALE Low to Valid Data In @16 MHz	t_{ALLVD}		8_{CLK} -150 $8t_{CLK}$ -90	ns ns
20	Valid Addr. to Valid Data In @16 MHz	t_{AVDV}		$9t_{CLK}$ -165 $9t_{CLK}$ -105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$ -50	$3t_{CLK}$ +50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$ -130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK} -60		ns
24	Data Valid to $\overline{WR}$ High @16 MHz	t_{DVWRH}	$7t_{CLK}$ -150 $7t_{CLK}$ -90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK} -50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK} -40	t_{CLK} +50	ns

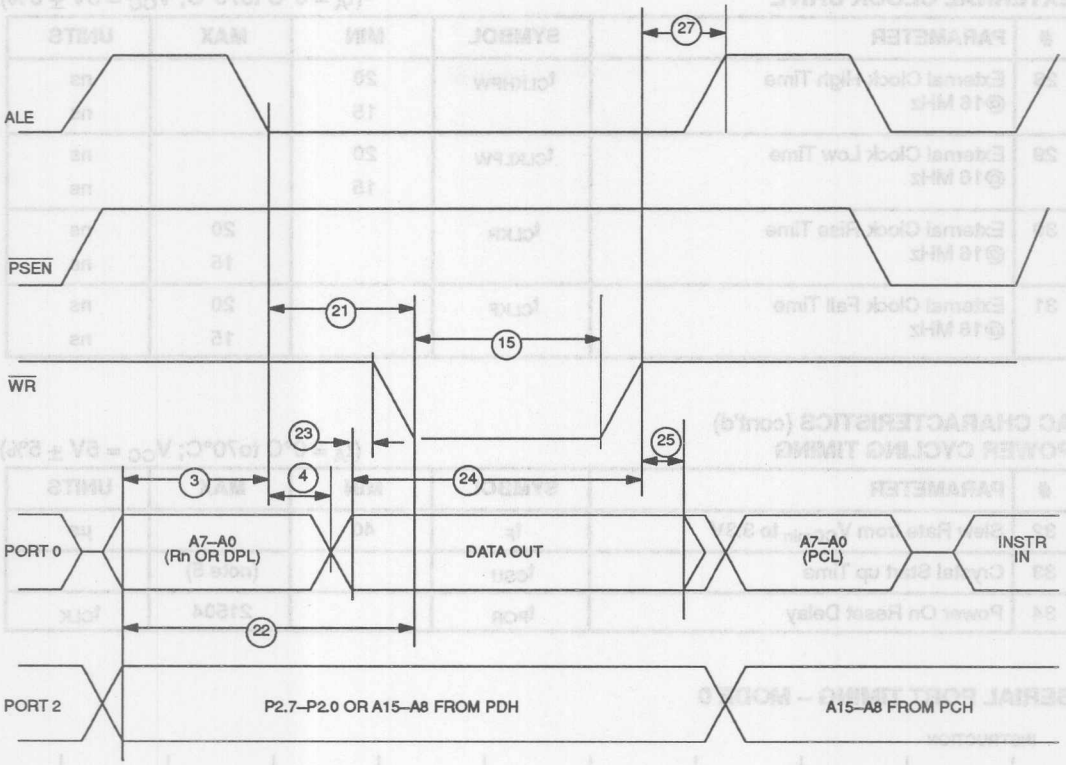
EXPANDED PROGRAM MEMORY READ CYCLE



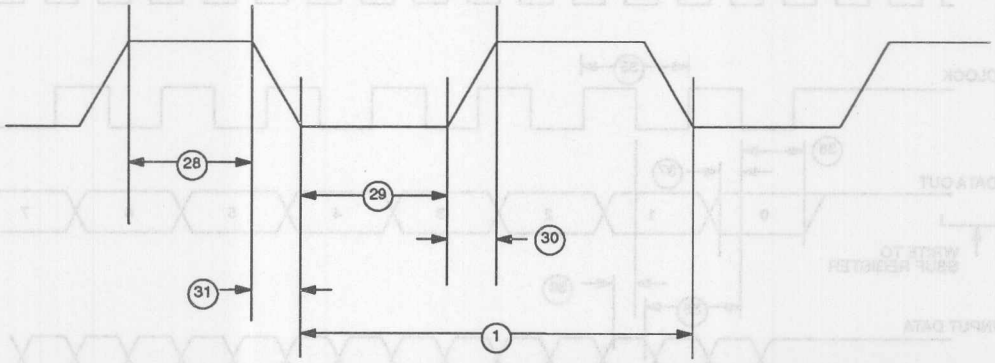
EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE



EXTERNAL CLOCK TIMING



AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

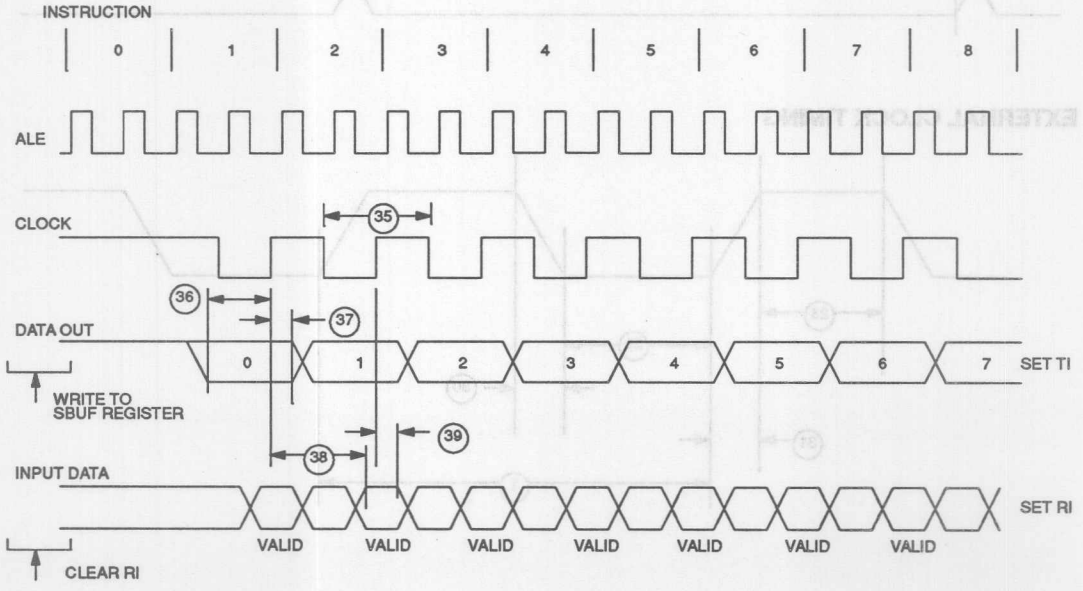
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @16 MHz	t_{CLKF}		20 15	ns ns

AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCmin} to 3.3V	t_F	40		µs
33	Crystal Start up Time	t_{CSU}		(note 5)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

SERIAL PORT TIMING – MODE 0

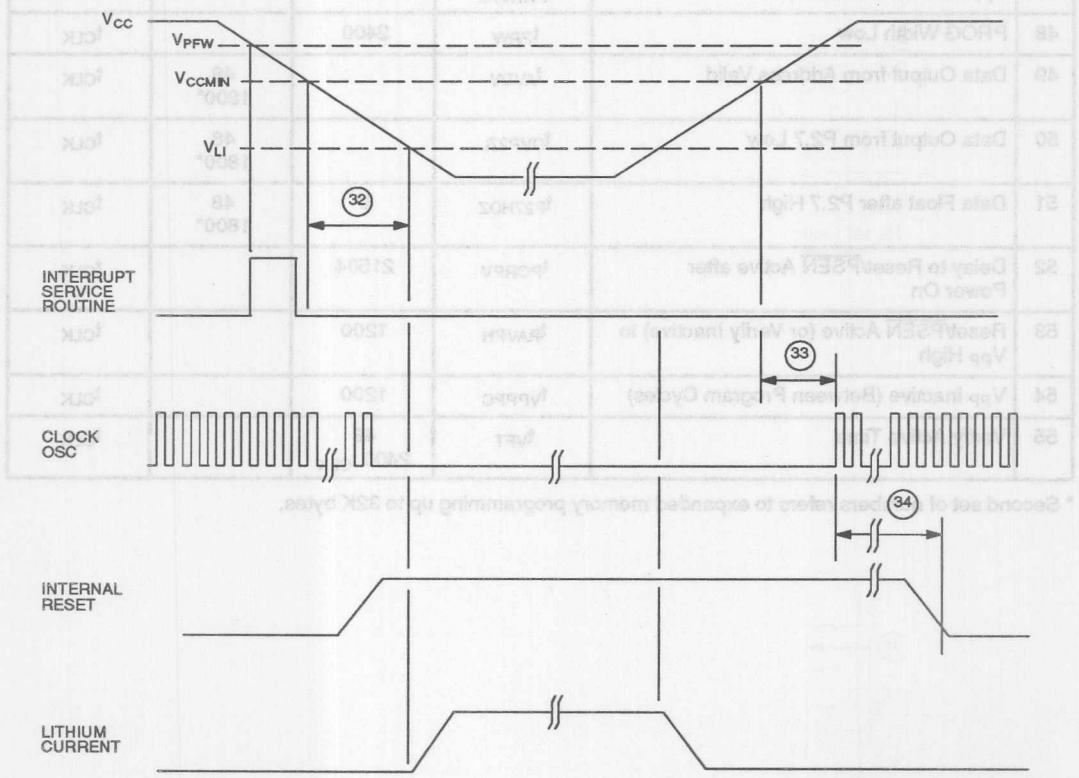


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$ -133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$ -117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$ -133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

POWER CYCLE TIMING



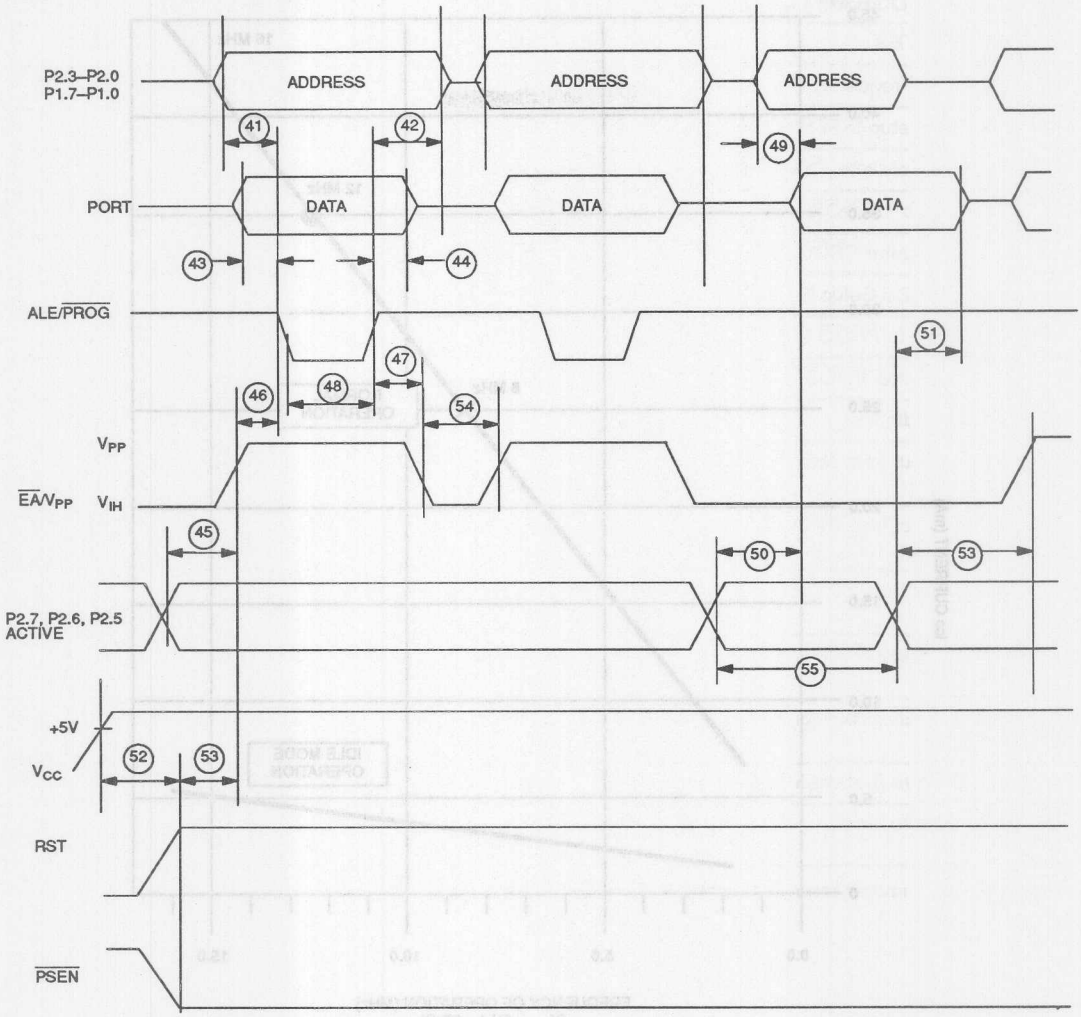
AC CHARACTERISTICS (cont'd)
PARALLEL PROGRAM LOAD TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Oscillator Frequency	$1/t_{CLK}$	1.0	12.0	MHz
41	Address Setup to $\overline{PROG}$ Low	t_{AVPRL}	0		
42	Address Hold after $\overline{PROG}$ High	t_{PRHAV}	0		
43	Data Setup to $\overline{PROG}$ Low	t_{DVPRL}	0		
44	Data Hold after $\overline{PROG}$ High	t_{PRHDV}	0		
45	P2.7, 2.6, 2.5 Setup to V_{PP}	t_{P27HVP}	0		
46	V_{PP} Setup to $\overline{PROG}$ Low	t_{VPHPRL}	0		
47	V_{PP} Hold after $\overline{PROG}$ Low	t_{PRHVPL}	0		
48	$\overline{PROG}$ Width Low	t_{PRW}	2400		t_{CLK}
49	Data Output from Address Valid	t_{AVDV}		48 1800*	t_{CLK}
50	Data Output from P2.7 Low	t_{DVP27L}		48 1800*	t_{CLK}
51	Data Float after P2.7 High	t_{P27HDZ}	0	48 1800*	t_{CLK}
52	Delay to Reset/$\overline{PSEN}$ Active after Power On	t_{PORPV}	21504		t_{CLK}
53	Reset/$\overline{PSEN}$ Active (or Verify Inactive) to V_{PP} High	t_{RAVPH}	1200		t_{CLK}
54	V_{PP} Inactive (Between Program Cycles)	t_{VPPPC}	1200		t_{CLK}
55	Verify Active Time	t_{VFT}	48 2400 t_{CLK}		t_{CLK}

* Second set of numbers refers to expanded memory programming up to 32K bytes.

PARALLEL PROGRAM LOAD TIMING

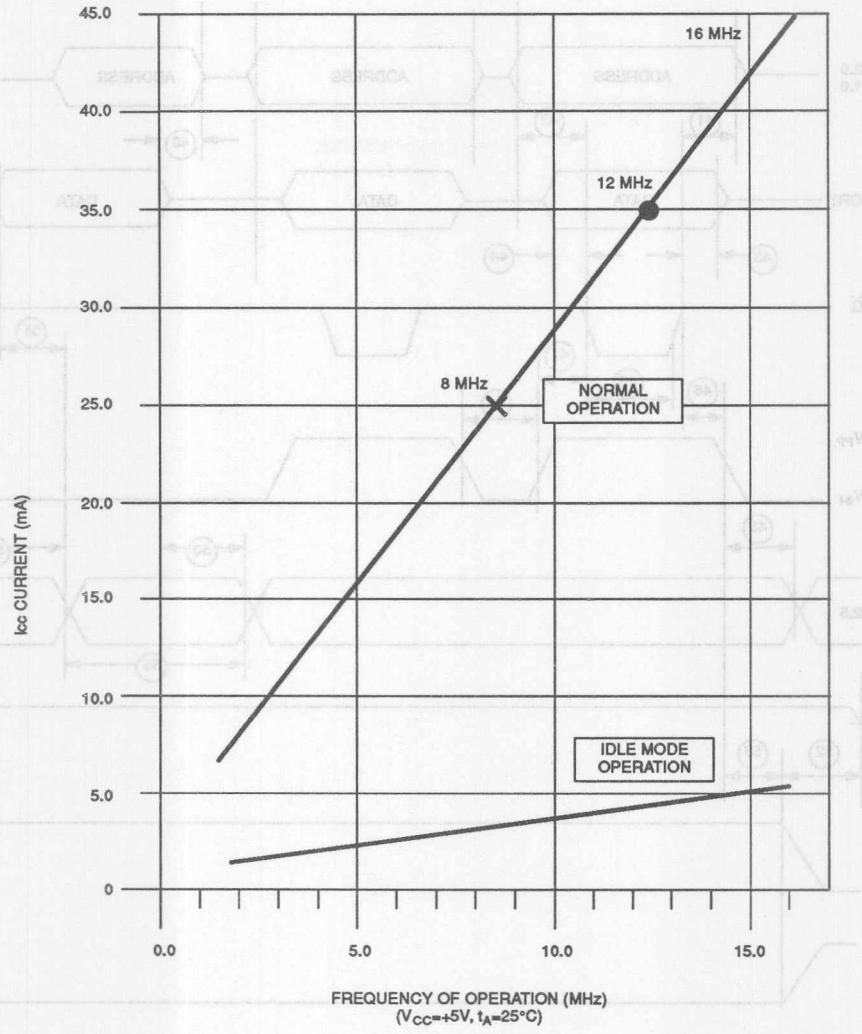


CAPACITANCE

(test frequency = 1 MHz; t_A = 25°C)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Output Capacitance	C_O			10	pF	
Input Capacitance	C_I			10	pF	

DS5000 TYPICAL I_{CC} VS. FREQUENCY



Normal operation is measured using:

1) External crystals on XTAL1 and 2
2) All port pins disconnected
3) RST=0 volts and EA=V_{CC}
4) Part performing endless loop writing to internal memory.

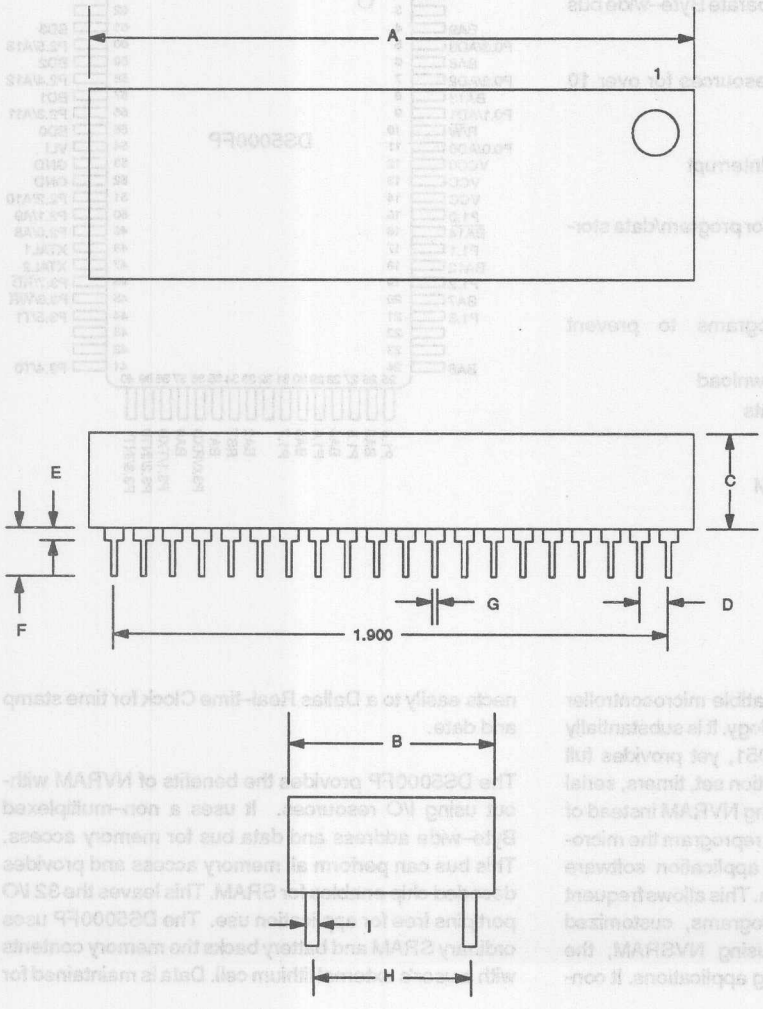
Idle mode operation is measured using:

1) External clock source at XTAL1; XTAL2 floating
2) All port pins disconnected
3) RST=0 volts and EA=V_{CC}
4) Part set in IDLE mode by software.

NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with tCLKR, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = RST = PORT0 = V_{CC}.
3. Idle mode I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = PORT0 = V_{CC}, RST = V_{SS}.
4. Stop mode I_{CC} is measured with all output pins disconnected; $\overline{EA}$ = PORT0 = V_{CC}; XTAL2 not connected; RST = V_{SS}.
5. Crystal start up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on-chip oscillator. The user should check with the crystal vendor for the worst case spec on this time.

PACKAGE DRAWING



DIM	INCHES	
	MIN	MAX
A IN.	2.080	2.100
B IN.	0.680	0.700
C IN.	0.290	0.310
D IN.	0.090	0.110
E IN.	0.040	0.060
F IN.	0.165	0.185
G IN.	0.016	0.020
H IN.	0.590	0.610
I IN.	0.009	0.012

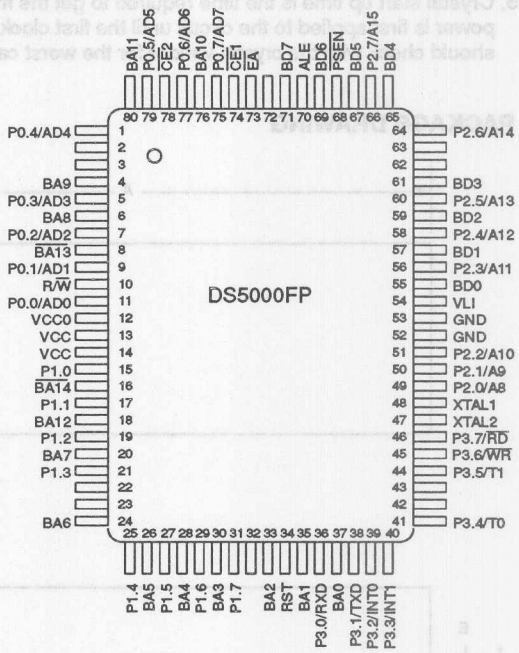
DALLAS SEMICONDUCTOR

DS5000FP
Soft Microcontroller Chip

FEATURES

- 8051 compatible uC adapts to its task
  - Accesses between 8K and 64K of nonvolatile SRAM
  - In–system programming via on–chip serial port
  - Can modifying its own program or data memory
  - Accesses memory on a separate Byte–wide bus
- Crashproof Operation
  - Maintains all nonvolatile resources for over 10 years
  - Power–fail Reset
  - Early Warning Power–fail Interrupt
  - Watchdog Timer
  - Lithium backs user SRAM for program/data storage
- Software Security
  - Executes encrypted programs to prevent observation
  - Security Lock prevents download
  - Unlocking destroys contents
- Fully 8051 Compatible
  - 128 bytes scratchpad RAM
  - Two timer/counters
  - On–chip serial port
  - 32 parallel I/O port pins

PIN ASSIGNMENT



DESCRIPTION

The DS5000FP is an 8051 compatible microcontroller based on nonvolatile RAM technology. It is substantially more flexible than a standard 8051, yet provides full compatibility with the 8051 instruction set, timers, serial port, and parallel I/O ports. By using NVRAM instead of ROM, the user can program, then reprogram the microcontroller while in–system. The application software can even change its own operation. This allows frequent software upgrades, adaptive programs, customized systems, etc. In addition, by using NVSRAM, the DS5000FP is ideal for data logging applications. It connects easily to a Dallas Real–time Clock for time stamp and date.

The DS5000FP provides the benefits of NVRAM without using I/O resources. It uses a non–multiplexed Byte–wide address and data bus for memory access. This bus can perform all memory access and provides decoded chip enables for SRAM. This leaves the 32 I/O port pins free for application use. The DS5000FP uses ordinary SRAM and battery backs the memory contents with a user's external lithium cell. Data is maintained for

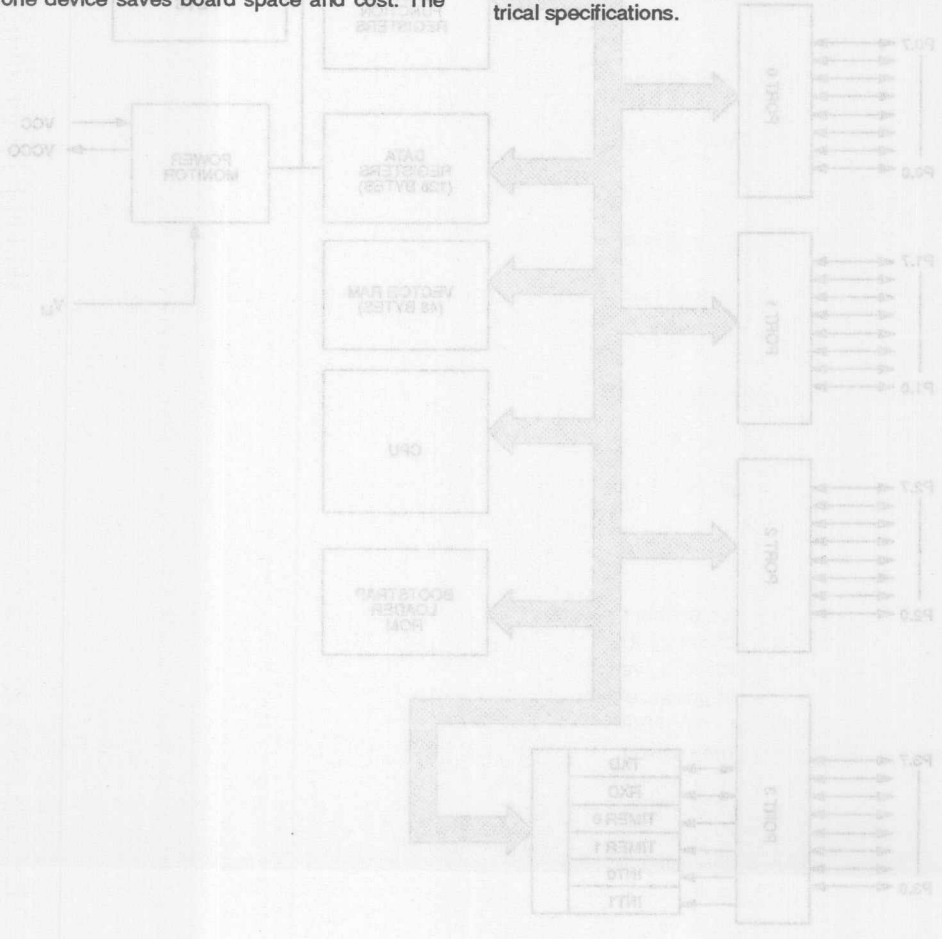
over 10 years with a very small lithium cell. A DS5000FP also provides crashproof operation in portable systems or systems with unreliable power. These features include the ability to save the operating state, Power–fail Reset, Power–fail Interrupt, and Watchdog Timer.

A user loads programs into the DS5000FP via its on–chip Serial Bootstrap Loader. This function supervises the loading of code into NVRAM, validates it, then becomes transparent to the user. Software can be stored in an 8K byte or 32K byte CMOS SRAM. Using its internal Partitioning, the DS5000FP will divide this common RAM into user programmable code and data segments. This Partition can be selected at program loading time, but can be modified anytime later. It will decode memory access to the SRAM, communicate via its Byte–wide bus and write–protect the memory portion designated as ROM. Combining program and data storage in one device saves board space and cost. The DS5000FP can also access a second 32K bytes of NVRAM but this area is restricted to data memory. For a user that wants a pre–constructed module using the DS5000FP, RAM, lithium cell, and optional clock; the DS2250(T) and DS5000(T) are available and described in separate data sheets. More details are also contained in the User's Guide section of the Soft Microcontroller Data Book.

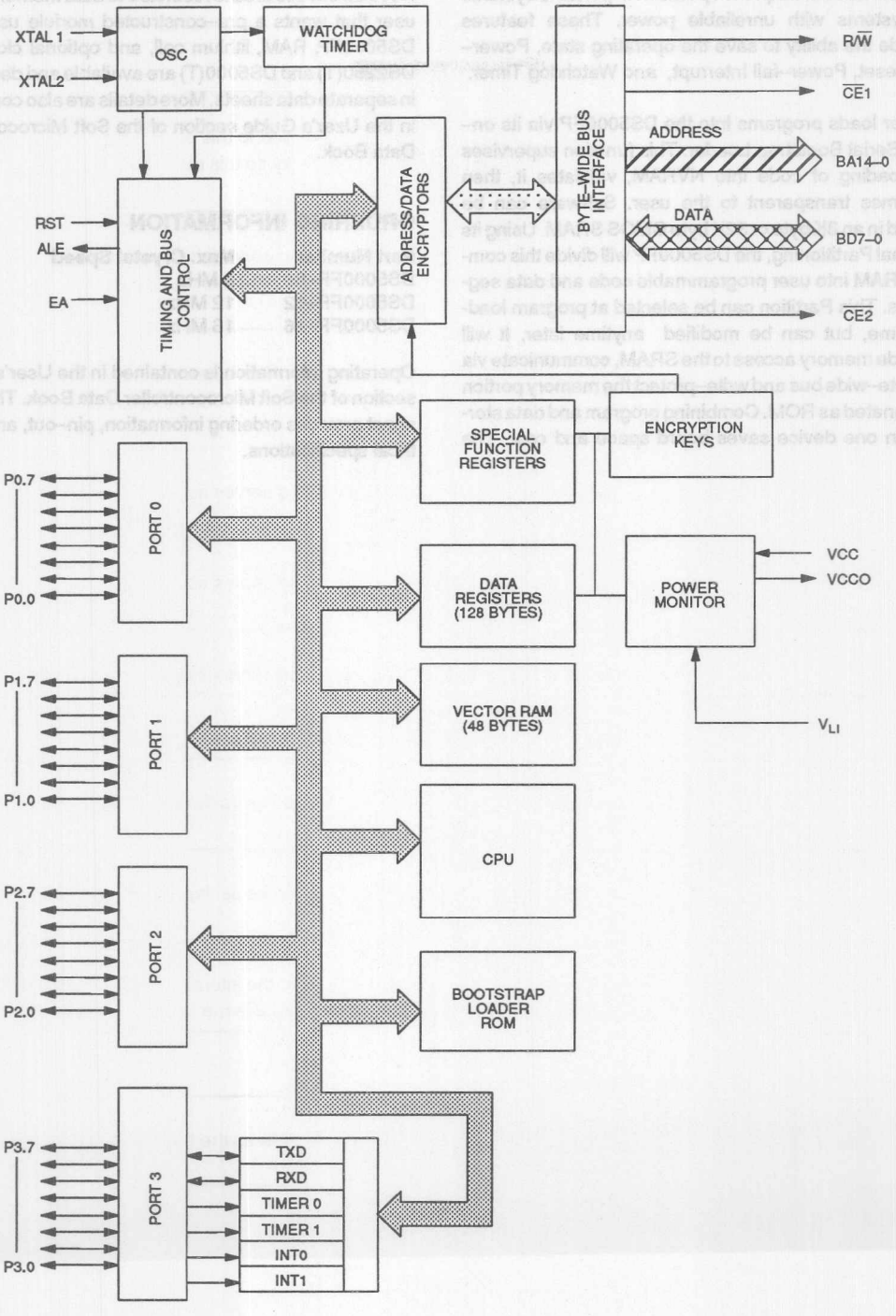
ORDERING INFORMATION

Part Number	Max. Crystal Speed
DS5000FP–8	8 MHz
DS5000FP–12	12 MHz
DS5000FP–16	16 MHz

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pin–out, and electrical specifications.



DS5000FP BLOCK DIAGRAM Figure 1



PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
15, 17, 19, 21, 25, 27, 29, 31	P1.0 – P1.7 General purpose I/O Port 1
34	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally so this pin can be left unconnected if not used.
36	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should not be connected directly to a PC COM port.
38	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should not be connected directly to a PC COM port.
39	P3.2 $\overline{\text{INT0}}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
40	P3.3 $\overline{\text{INT1}}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
41	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
44	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
45	P3.6 $\overline{\text{WR}}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
46	P3.7 $\overline{\text{RD}}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
47, 48	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
52, 53	GND Logic ground.
49, 50, 51, 56, 58, 60, 64, 66	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.

PIN NUMBER	DESCRIPTION
68	$\overline{PSEN}$ Program Store Enable. This active low signal is used to enable an external program memory when using the Expanded bus. It is normally an output and should be unconnected if not used. $\overline{PSEN}$ also is used to invoke the Bootstrap Loader. At this time, $\overline{PSEN}$ will be pulled down externally. This should only be done once the DS5000 is already in a reset state. The device that pulls down should be open drain since it must not interfere with $\overline{PSEN}$ under normal operation.
70	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch. When using a parallel programmer, this pin also assumes the $\overline{PROG}$ function for programming pulses.
73	$\overline{EA}$ External Access. This pin forces the DS5000 to behave like an 8031. No internal memory (or clock) will be available when this pin is at a logic low. Since this pin is pulled down internally, it should be connected to +5V to use NVRAM. In a parallel programmer, this pin also serves as V_{PP} for super voltage pulses.
11, 9, 7, 5, 1, 79, 77, 75	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull– ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
13, 14	V_{CC} +5V
16, 8, 18, 80, 76, 4, 6, 20, 24, 26, 28, 30, 33, 35, 37	BA14–0 Byte–wide Address bus bits 14–0. This 15 bit bus is combined with the non–multiplexed data bus (BD7–0) to access NVSRAM. Decoding is performed on $\overline{CE1}$ and $\overline{CE2}$. Read/write access is controlled by R/$\overline{W}$. BA14–0 connect directly to an 8K or 32K SRAM. If an 8K RAM is used, BA13 and BA14 will be unconnected. Note BA13 and BA14 are inverted from the true logical address. Also note that BA14 is lithium backed.
71, 69, 67, 65, 61, 59, 57, 55	BD7–0 Byte–wide Data bus bits 7–0. This 8 bit bi–directional bus is combined with the non–multiplexed address bus (BA14–0) to access NVSRAM. Decoding is performed on $\overline{CE1}$ and $\overline{CE2}$. Read/write access is controlled by R/$\overline{W}$. BD7–0 connect directly to an 8K or 32K SRAM, and optionally to a Real–time Clock.
10	R/$\overline{W}$ Read/Write. This signal provides the write enable to the SRAMs on the Byte–wide bus. It is controlled by the memory map and Partition. The blocks selected as Program (ROM) will be write protected.

PIN NUMBER	DESCRIPTION
74	$\overline{CE1}$ Chip Enable 1. This is the primary decoded chip enable for memory access on the Byte–wide bus. It connects to the chip enable input of one SRAM. $\overline{CE1}$ is lithium backed. It will remain in a logic high inactive state when V_{CC} falls below V_{LI}.
78	$\overline{CE2}$ Chip Enable 2. This chip enable is provided to bank switch to a second block of 32K bytes of nonvolatile data memory. It connects to the chip enable input of one SRAM or one lithium backed peripheral such a DS1283 clock. $\overline{CE2}$ is lithium backed. It will remain in a logic high inactive state when V_{CC} falls below V_{LI}.
12	V_{CCO} V_{CC} Output. This is switched between V_{CC} and V_{LI} by internal circuits based on the level of V_{CC}. When power is above the lithium input, power will be drawn from V_{CC}. The lithium cell remains isolated form a load. When V_{CC} is below V_{LI}, the V_{CCO} switches to the V_{LI} source. V_{CCO} is connected to the V_{CC} pin of an SRAM.
54	V_{LI} Lithium Voltage Input. Connect to a lithium cell greater than V_{LImin} and no greater than V_{LImax} as shown in the electrical specifications. Nominal value is +3V.

INSTRUCTION SET

The DS5000FP executes an instruction set that is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages such as assemblers and compilers that have been written for the 8051 are compatible with the DS5000FP. A complete description of the instruction set and operation are provided in the User's Guide section of the Soft Microcontroller Data Book.

Also note that the DS5000FP is embodied in the DS5000(T) and DS2250(T) modules. The DS5000(T) combines the DS5000FP with one SRAM of either 8K or 32K bytes and a lithium cell. An optional Real–time Clock is also available in the DS5000T. This is packaged in a 40–pin DIP module. The DS2250(T) is an identical function in a SIMM form factor. It also offers the option of a second 32K SRAM mapped as data on Chip Enable 2.

MEMORY ORGANIZATION

Figure 2 illustrates the memory map accessed by the DS5000FP. The entire 64K of program and 64K of data is available. The DS5000FP maps 32K of this space into the SRAM connected to the Byte–wide bus. This is the area from 0000h to 7FFFh (32K) and is reached via $\overline{CE1}$. Any area not mapped into the NVRAM is reached via the Expanded bus on Ports 0 & 2. Selecting $\overline{CE2}$ provides another 32K of potential data storage. When $\overline{CE2}$ is used, no data is available on the ports. The memory map is covered in detail in the User's Guide section of the Soft Microcontroller Data Book.

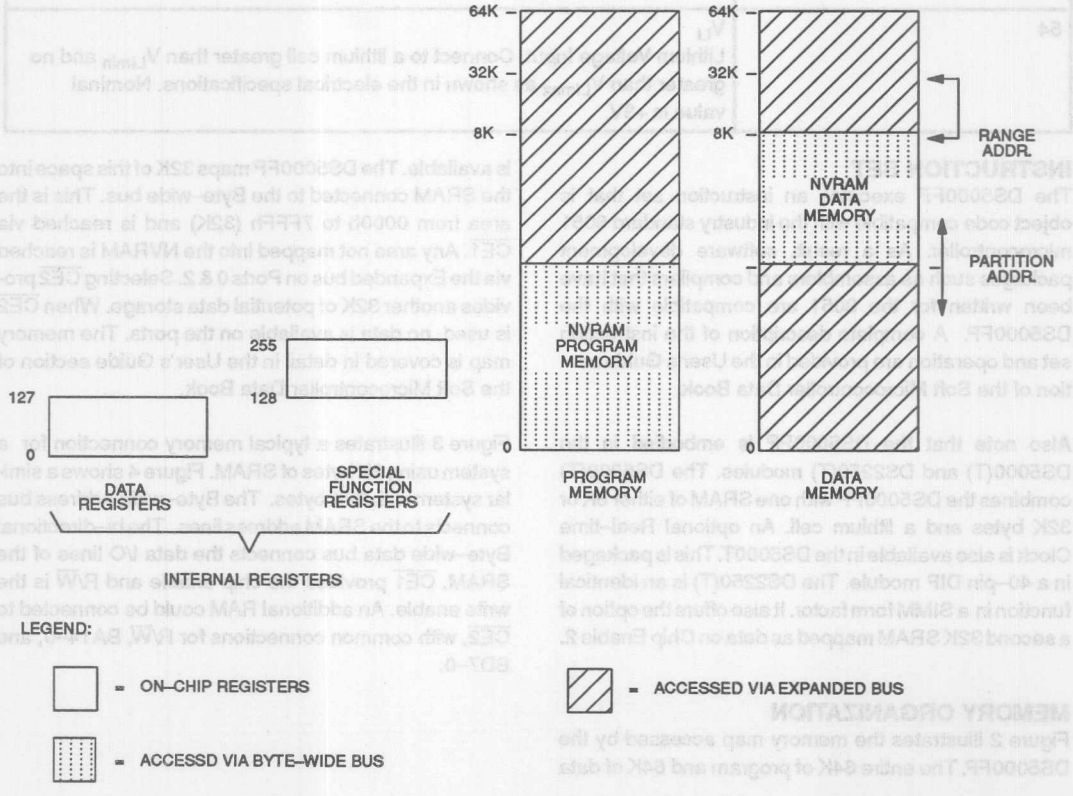
Figure 3 illustrates a typical memory connection for a system using 8K bytes of SRAM. Figure 4 shows a similar system with 32K bytes. The Byte–wide Address bus connects to the SRAM address lines. The bi–directional Byte–wide data bus connects the data I/O lines of the SRAM. $\overline{CE1}$ provides the chip enable and R/$\overline{W}$ is the write enable. An additional RAM could be connected to $\overline{CE2}$, with common connections for R/$\overline{W}$, BA14–0, and BD7–0.

POWER MANAGEMENT

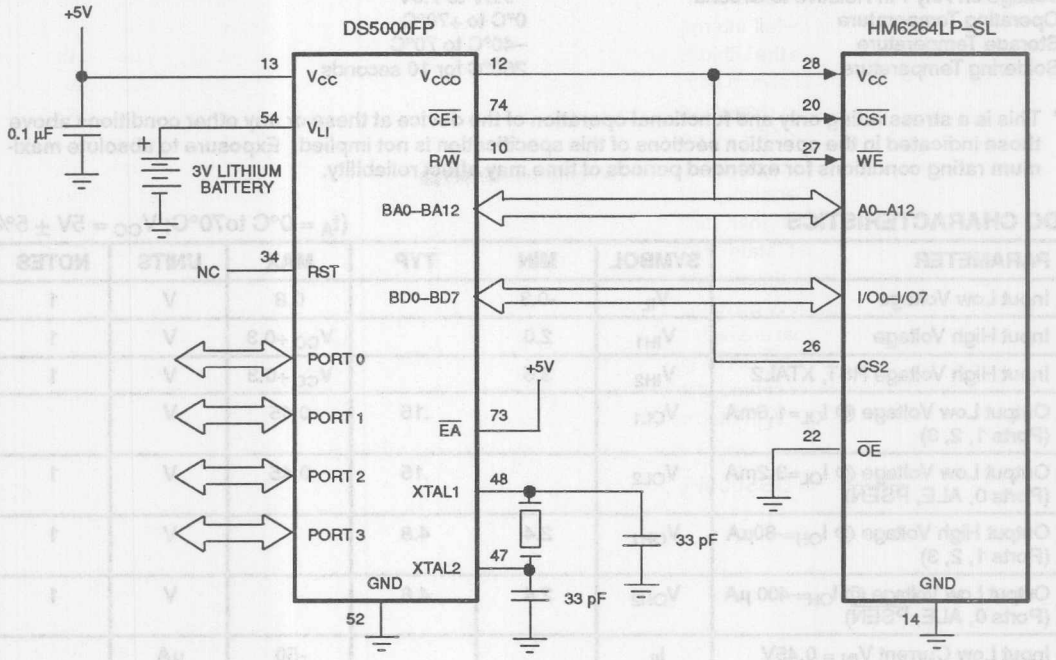
The DS5000FP monitors power to provide Power–fail Reset, early warning Power–fail Interrupt, and switch over to lithium backup. It uses the Lithium cell at V_{LI} as a reference in determining the switch points. These are called V_{PFW}, V_{CCMIN}, and V_{LI} respectively. When V_{CC} drops below V_{PFW}, the DS5000FP will perform an interrupt vector to location 2Bh if the power fail warning was enabled. Full processor operation continues regardless. When power falls further to V_{CCMIN}, the DS5000FP invokes a reset state. No further code execution will be performed unless power rises back above V_{CCMIN}. $\overline{CE1}$, $\overline{CE2}$, R/$\overline{W}$ go to an inactive (logic 1) state. Any address lines that are high (due to encryption) will follow V_{CC}, except for BA14 which is lithium backed. V_{CC} is still the power source at this time. When V_{CC} drops further to below V_{LI}, internal circuitry will switch to the lithium cell for power. The majority of internal circuits will be disabled and the remaining nonvolatile states will be retained. Any devices connected to V_{CCO} will be powered by the lithium cell at this time. V_{CCO} will be at the lithium battery voltage less a diode drop. This drop will vary depending on the load. Low leakage SRAMs should be used for this reason. When a module is used, the lithium cell is selected by Dallas so absolute specifications are provided for the switch thresholds. When using the DS5000FP, the user must select the appropriate battery. The following formulas apply to the switch function.

$V_{PFW} = 1.45 * V_{LI}$
$V_{CCMIN} = 1.40 * V_{LI}$
V_{LI} Switch $= 1.0 * V_{LI}$

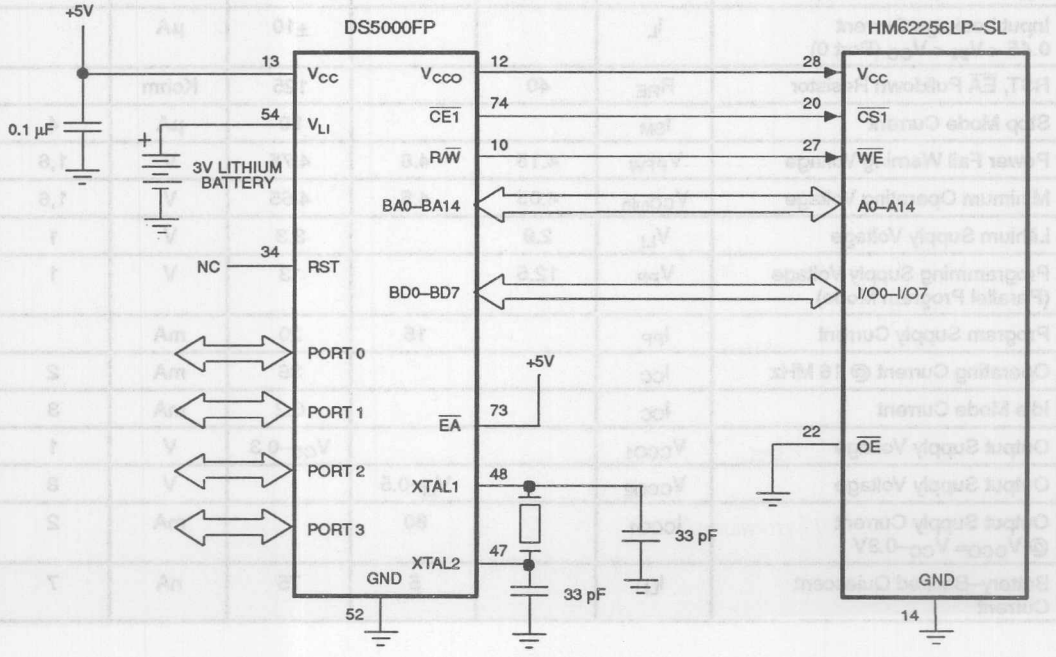
MEMORY MAP OF THE DS5000FP Figure 2



DS5000FP CONNECTION TO 8K X 8 SRAM Figure 3



DS5000FP CONNECTION TO 32K X 8 SRAM Figure 4



ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

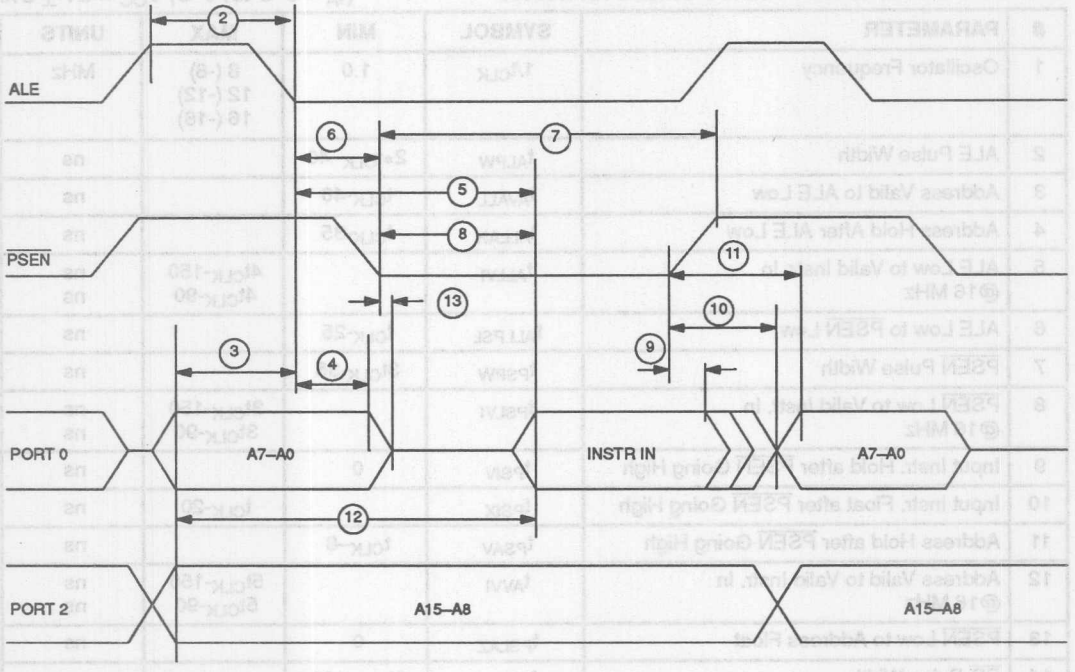
DC CHARACTERISTICS

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

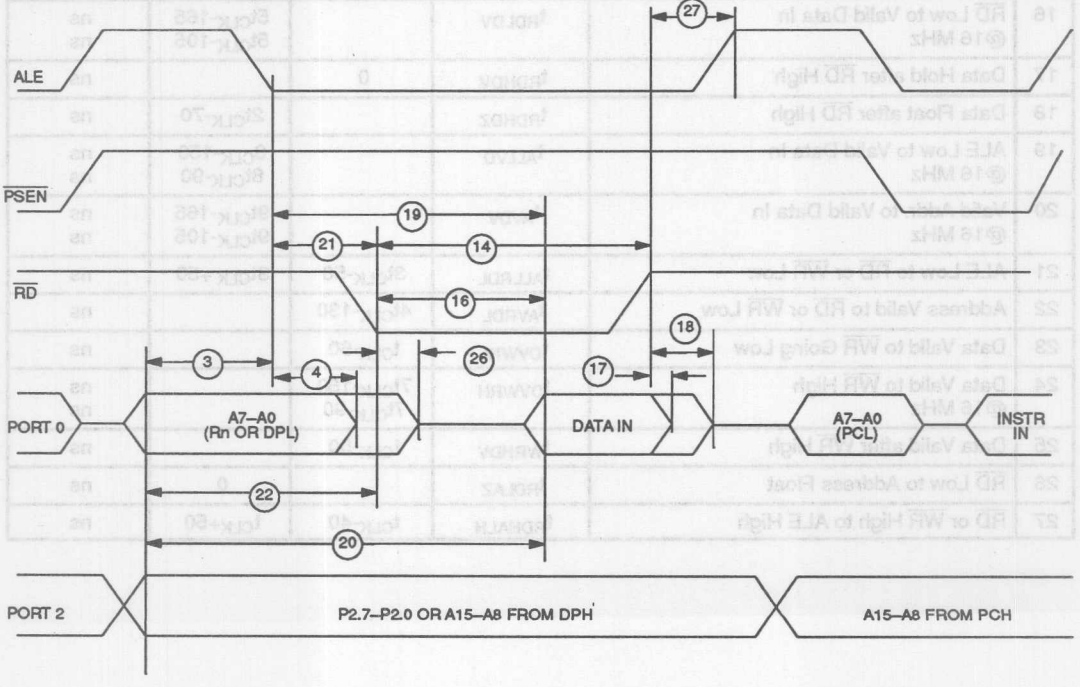
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	-0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC} +0.3	V	1
Input High Voltage RST, XTAL2	V_{IH2}	3.5		V_{CC} +0.3	V	1
Output Low Voltage @ I_{OL}=1.6mA (Ports 1, 2, 3)	V_{OL1}		.15	0.45	V	
Output Low Voltage @ I_{OL}=3.2mA (Ports 0, ALE, $\overline{PSEN}$)	V_{OL2}		.15	0.45	V	1
Output High Voltage @ I_{OH}=-80µA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output Low Voltage @ I_{OH}=-400 µA (Ports 0, ALE, $\overline{PSEN}$)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN} = 0.45V (Ports 1, 2, 3)	I_{IL}			-50	µA	
Transition Current; 1 to 0 V_{IN} = 2.0V (Ports 1, 2, 3)	I_{TL}			-500	µA	
Input Leakage Current 0.45 < V_{IN} < V_{CC} (Port 0)	I_L			±10	µA	
RST, $\overline{EA}$ Pulldown Resistor	R_{RE}	40		125	Kohm	
Stop Mode Current	I_{SM}			80	µA	4
Power Fail Warning Voltage	V_{PFW}	4.15	4.6	4.75	V	1,6
Minimum Operating Voltage	V_{CCmin}	4.05	4.5	4.65	V	1,6
Lithium Supply Voltage	V_{LI}	2.9		3.3	V	1
Programming Supply Voltage (Parallel Program Mode)	V_{PP}	12.5		13	V	1
Program Supply Current	I_{PP}		15	20	mA	
Operating Current @ 16 MHz	I_{CC}			36	mA	2
Idle Mode Current	I_{CC}			6.2	mA	3
Output Supply Voltage	V_{CCO1}			V_{CC}–0.3	V	1
Output Supply Voltage	V_{CCO2}		V_{LI}–0.5		V	8
Output Supply Current @ V_{CCO}= V_{CC}–0.3V	I_{CCO1}		80		mA	2
Battery–Backed Quiescent Current	I_{LI}		5	75	nA	7

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	8 (-8) 12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2*t_{CLK}-40$		ns
3	Address Valid to ALE Low	t_{AVALL}	$t_{CLK}-40$		ns
4	Address Hold After ALE Low	t_{ALLAH}	$t_{CLK}-35$		ns
5	ALE Low to Valid Instr. In @16 MHz	t_{ALLVI}		$4t_{CLK}-150$ $4t_{CLK}-90$	ns ns
6	ALE Low to $\overline{PSEN}$ Low	t_{ALLPSL}	$t_{CLK}-25$		ns
7	$\overline{PSEN}$ Pulse Width	t_{PSPW}	$3t_{CLK}-35$		ns
8	$\overline{PSEN}$ Low to Valid Instr. In @16 MHz	t_{PSLVI}		$3t_{CLK}-150$ $3t_{CLK}-90$	ns ns
9	Input Instr. Hold after $\overline{PSEN}$ Going High	t_{PSIV}	0		ns
10	Input Instr. Float after $\overline{PSEN}$ Going High	t_{PSIX}		$t_{CLK}-20$	ns
11	Address Hold after $\overline{PSEN}$ Going High	t_{PSAV}	$t_{CLK}-8$		ns
12	Address Valid to Valid Instr. In @16 MHz	t_{AVVI}		$5t_{CLK}-150$ $5t_{CLK}-90$	ns ns
13	$\overline{PSEN}$ Low to Address Float	t_{PSLAZ}	0		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}-100$		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}-100$		ns
16	$\overline{RD}$ Low to Valid Data In @16 MHz	t_{RDLDV}		$5t_{CLK}-165$ $5t_{CLK}-105$	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}-70$	ns
19	ALE Low to Valid Data In @16 MHz	t_{ALLVD}		$8_{CLK}-150$ $8t_{CLK}-90$	ns ns
20	Valid Addr. to Valid Data In @16 MHz	t_{AVDV}		$9t_{CLK}-165$ $9t_{CLK}-105$	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}-50$	$3t_{CLK}+50$	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}-130$		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	$t_{CLK}-60$		ns
24	Data Valid to $\overline{WR}$ High @16 MHz	t_{DVWRH}	$7t_{CLK}-150$ $7t_{CLK}-90$		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	$t_{CLK}-50$		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	$t_{CLK}-40$	$t_{CLK}+50$	ns

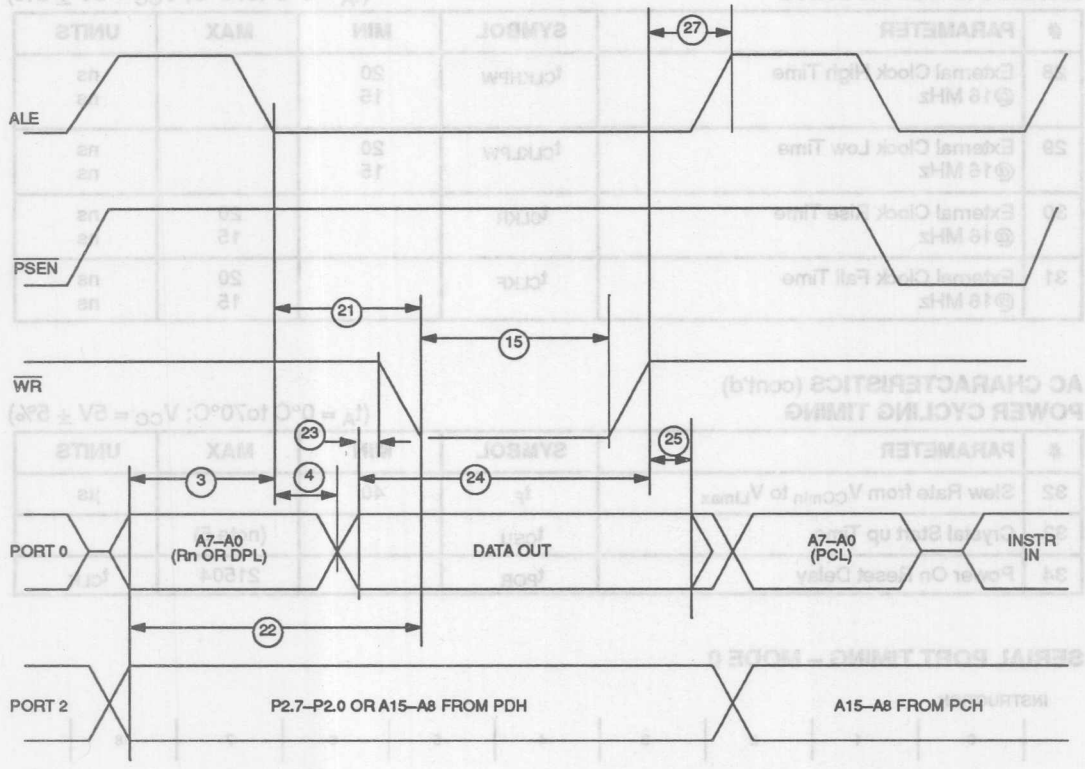
EXPANDED PROGRAM MEMORY READ CYCLE



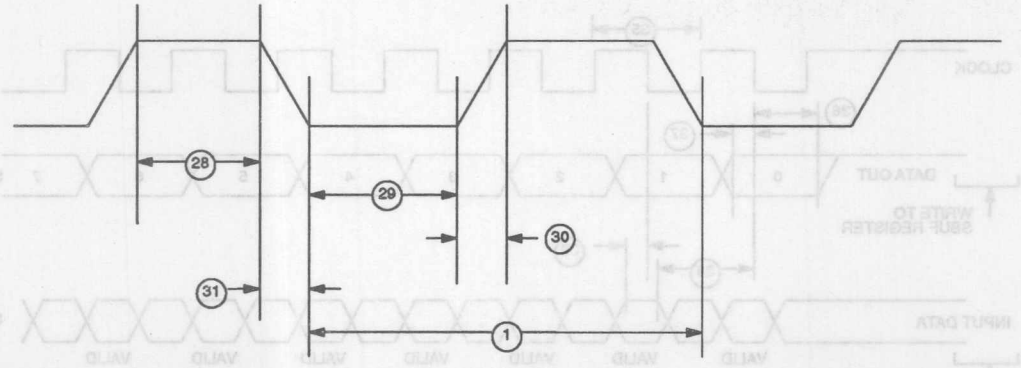
EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE



EXTERNAL CLOCK TIMING



AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

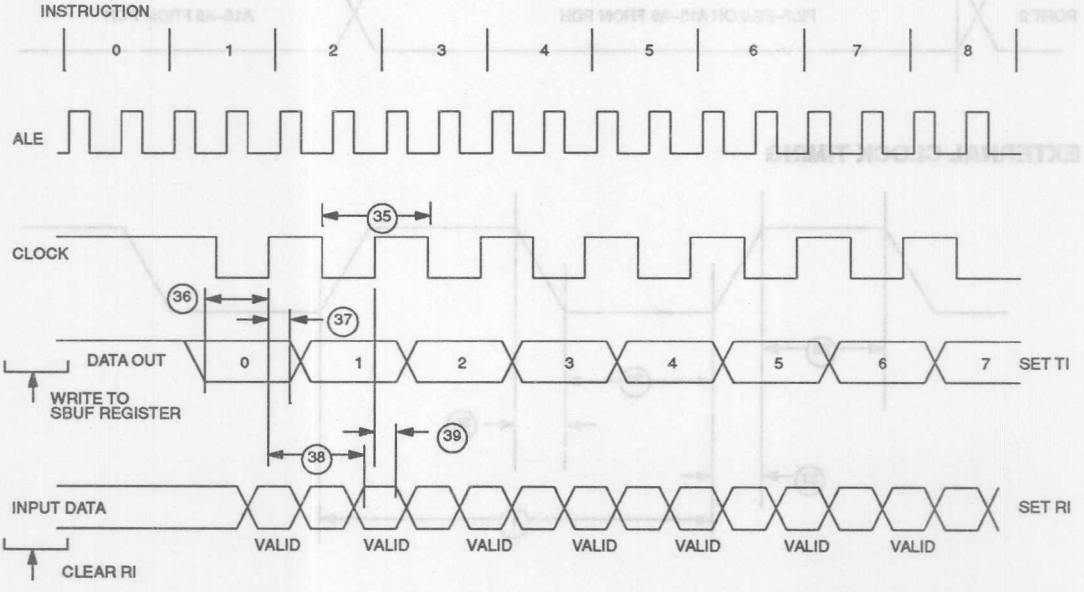
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @16 MHz	t_{CLKF}		20 15	ns ns

AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCmin} to V_{LImax}	t_F	40		µs
33	Crystal Start up Time	t_{CSU}		(note 5)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

SERIAL PORT TIMING – MODE 0

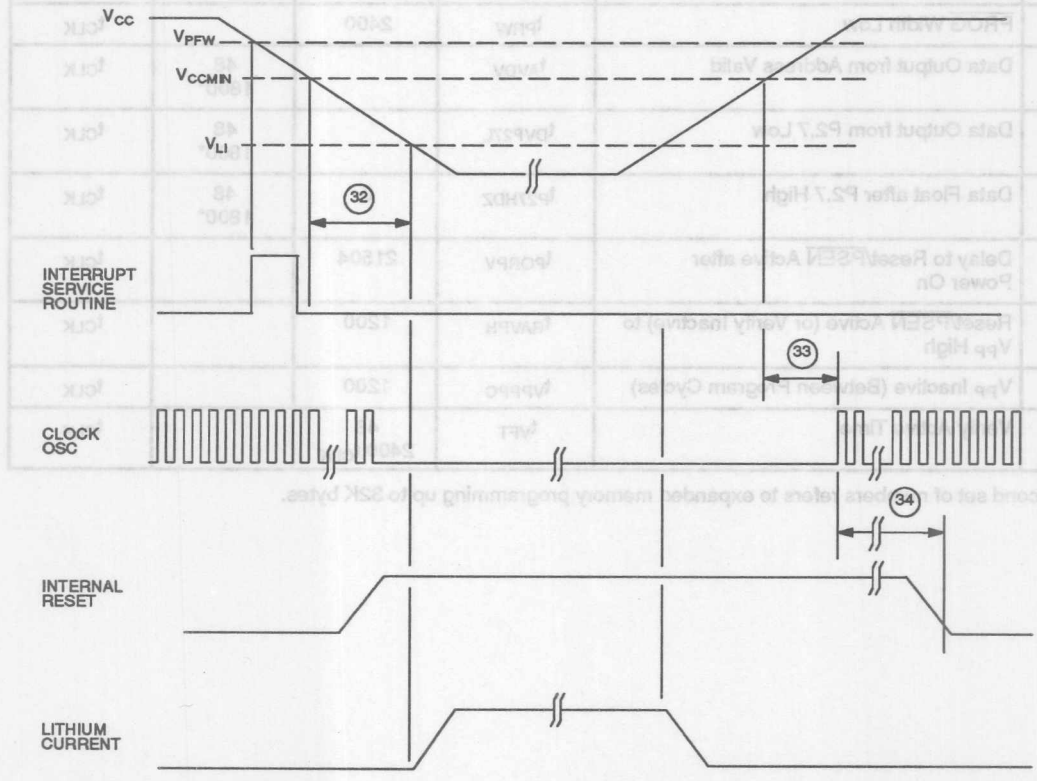


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$-133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$-117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$-133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

POWER CYCLE TIMING



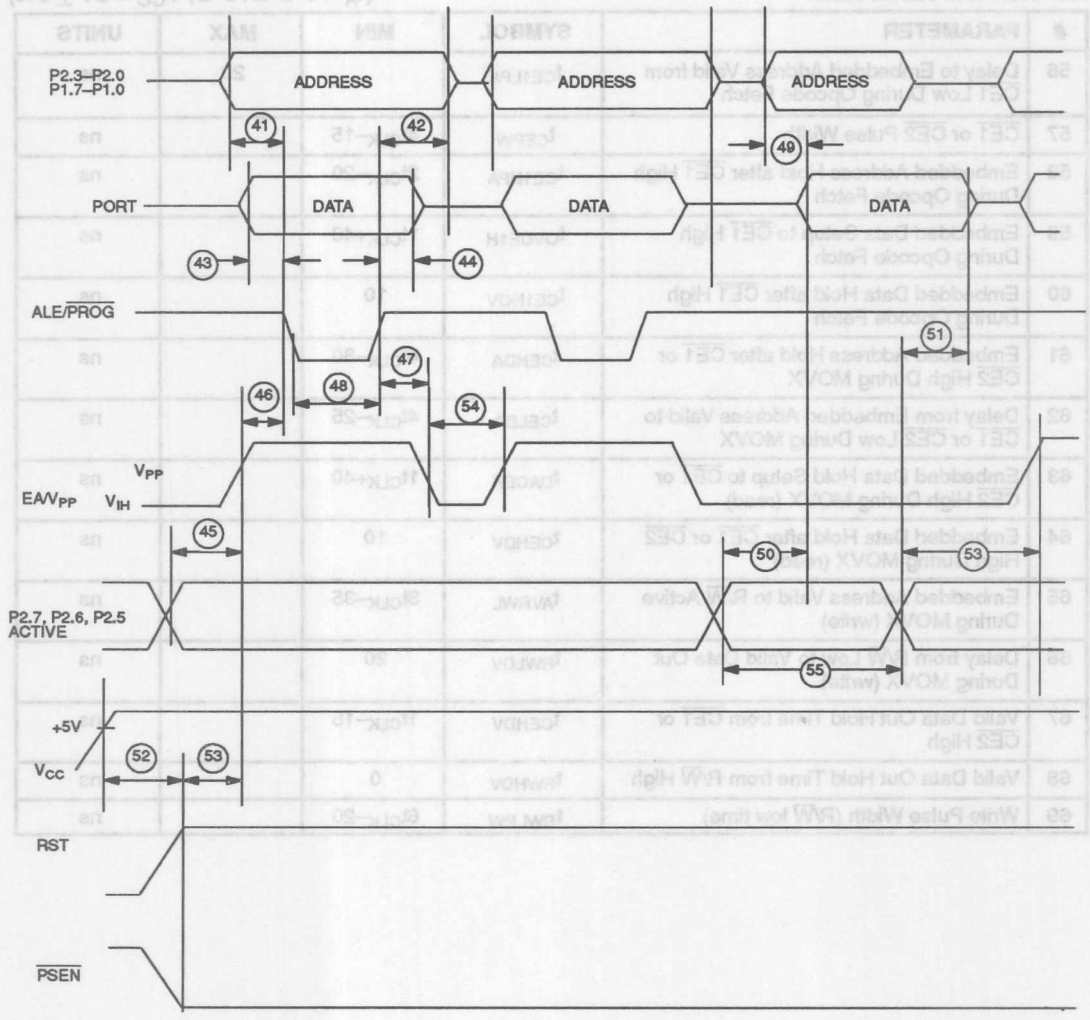
AC CHARACTERISTICS (cont'd)
PARALLEL PROGRAM LOAD TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Oscillator Frequency	$1/t_{CLK}$	1.0	12.0	MHz
41	Address Setup to $\overline{PROG}$ Low	t_{AVPRL}	0		
42	Address Hold after $\overline{PROG}$ High	t_{PRHAV}	0		
43	Data Setup to $\overline{PROG}$ Low	t_{DVPRL}	0		
44	Data Hold after $\overline{PROG}$ High	t_{PRHDV}	0		
45	P2.7, 2.6, 2.5 Setup to V_{PP}	t_{P27HVP}	0		
46	V_{PP} Setup to $\overline{PROG}$ Low	t_{VPHPRL}	0		
47	V_{PP} Hold after $\overline{PROG}$ Low	t_{PRHVPL}	0		
48	$\overline{PROG}$ Width Low	t_{PRW}	2400		t_{CLK}
49	Data Output from Address Valid	t_{AVDV}		48 1800*	t_{CLK}
50	Data Output from P2.7 Low	t_{DVP27L}		48 1800*	t_{CLK}
51	Data Float after P2.7 High	t_{P27HDZ}	0	48 1800*	t_{CLK}
52	Delay to Reset/$\overline{PSEN}$ Active after Power On	t_{PORPV}	21504		t_{CLK}
53	Reset/$\overline{PSEN}$ Active (or Verify Inactive) to V_{PP} High	t_{RAVPH}	1200		t_{CLK}
54	V_{PP} Inactive (Between Program Cycles)	t_{VPPPC}	1200		t_{CLK}
55	Verify Active Time	t_{VFT}	48 2400 t_{CLK}		t_{CLK}

* Second set of numbers refers to expanded memory programming up to 32K bytes.

PARALLEL PROGRAM LOAD TIMING



CAPACITANCE

(test frequency = 1 MHz; t_A = 25°C)

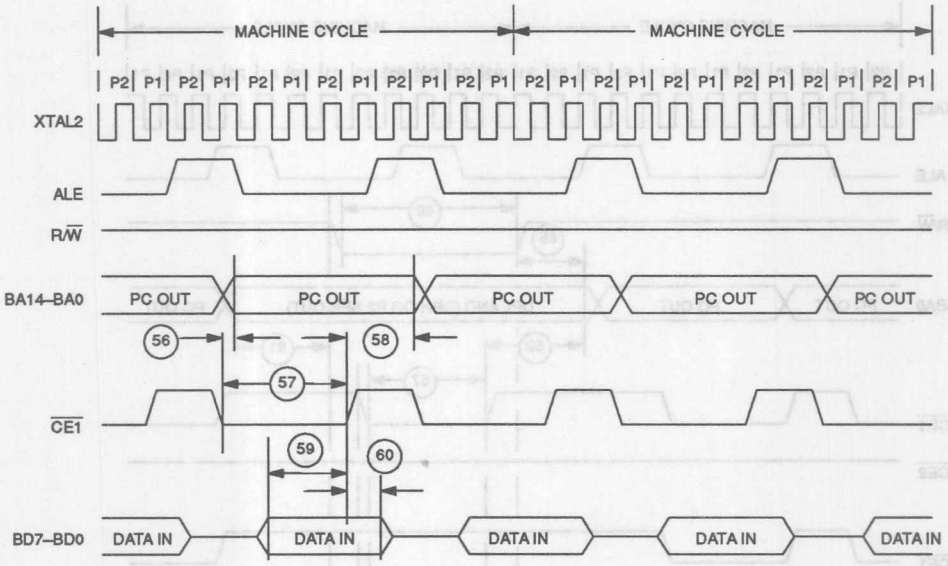
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Output Capacitance	C_O			10	pF	
Input Capacitance	C_I			10	pF	

BYTE–WIDE ADDRESS/DATA BUS TIMING AC CHARACTERISTICS

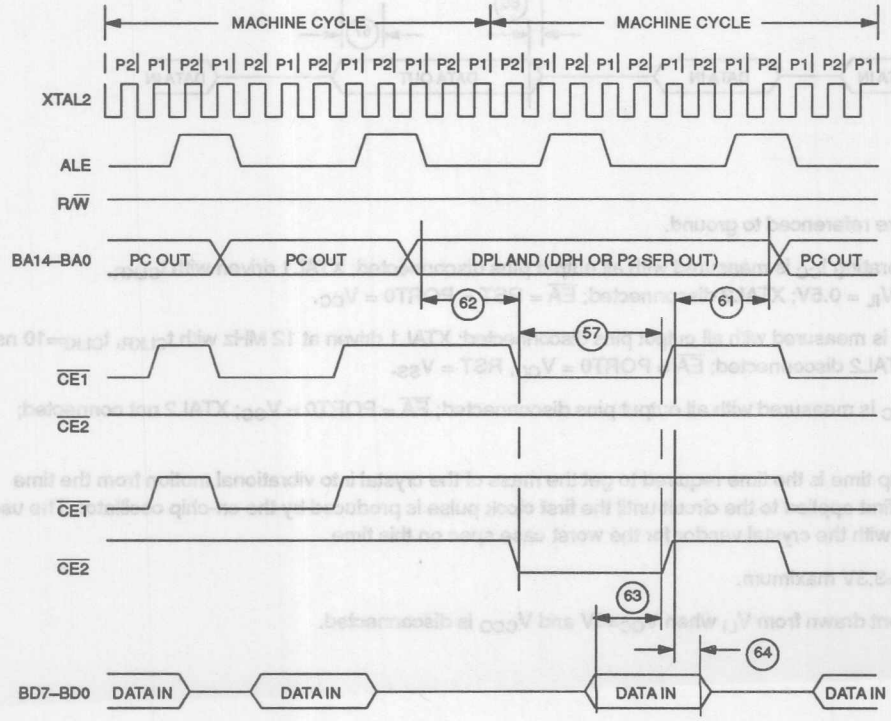
(t_A = 0°C to70°C; V_{CC} = 5V ± 5%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
56	Delay to Embedded Address Valid from $\overline{CE1}$ Low During Opcode Fetch	t_{CE1LPA}		20	ns
57	$\overline{CE1}$ or $\overline{CE2}$ Pulse Width	t_{CEPW}	$4t_{CLK}$–15		ns
58	Embedded Address Hold after $\overline{CE1}$ High During Opcode Fetch	t_{CE1HPA}	$2t_{CLK}$–20		ns
59	Embedded Data Setup to $\overline{CE1}$ High During Opcode Fetch	t_{OVCE1H}	$1t_{CLK}$+40		ns
60	Embedded Data Hold after $\overline{CE1}$ High During Opcode Fetch	t_{CE1HOV}	10		ns
61	Embedded Address Hold after $\overline{CE1}$ or $\overline{CE2}$ High During MOVX	t_{CEHDA}	$4t_{CLK}$–30		ns
62	Delay from Embedded Address Valid to $\overline{CE1}$ or $\overline{CE2}$ Low During MOVX	t_{CELDA}	$4t_{CLK}$–25		ns
63	Embedded Data Hold Setup to $\overline{CE1}$ or $\overline{CE2}$ High During MOVX (read)	t_{DACEH}	$1t_{CLK}$+40		ns
64	Embedded Data Hold after $\overline{CE1}$ or $\overline{CE2}$ High During MOVX (read)	t_{CEHDV}	10		ns
65	Embedded Address Valid to R/$\overline{W}$ Active During MOVX (write)	t_{AVRWL}	$3t_{CLK}$–35		ns
66	Delay from R/$\overline{W}$ Low to Valid Data Out During MOVX (write)	t_{RWLDV}	20		ns
67	Valid Data Out Hold Time from $\overline{CE1}$ or $\overline{CE2}$ High	t_{CEHDV}	$1t_{CLK}$–15		ns
68	Valid Data Out Hold Time from R/$\overline{W}$ High	t_{RWHDV}	0		ns
69	Write Pulse Width (R/$\overline{W}$ low time)	t_{RWLPW}	$6t_{CLK}$–20		ns

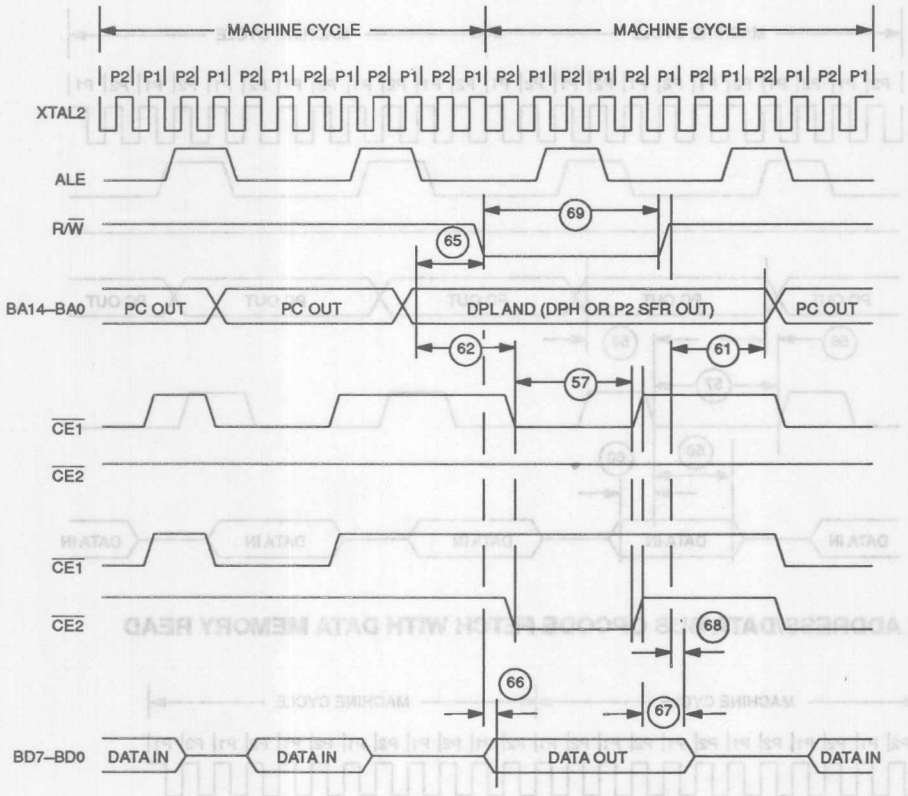
BYTE–WIDE ADDRESS/DATA BUS OPCODE FETCH CYCLE



BYTE–WIDE ADDRESS/DATA BUS OPCODE FETCH WITH DATA MEMORY READ



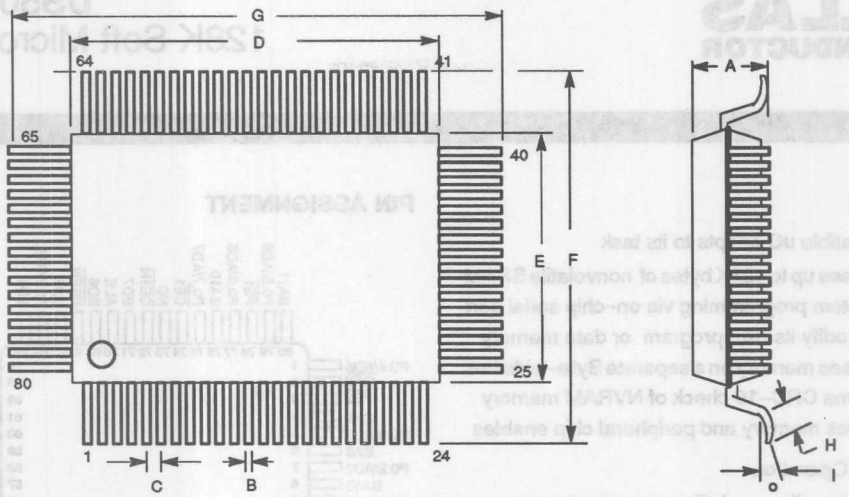
BYTE–WIDE ADDRESS/DATA BUS OPCODE FETCH WITH DATA MEMORY WRITE



NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = RST = PORT0 = V_{CC}.
3. Idle mode I_{CC} is measured with all output pins disconnected; XTAL1 driven at 12 MHz with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; $\overline{EA}$ = PORT0 = V_{CC}, RST = V_{SS}.
4. Stop mode I_{CC} is measured with all output pins disconnected; $\overline{EA}$ = PORT0 = V_{CC}; XTAL2 not connected; RST = V_{SS}.
5. Crystal start up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on-chip oscillator. The user should check with the crystal vendor for the worst case spec on this time.
6. Assumes V_{LI}=3.3V maximum.
7. I_{LI} is the current drawn from V_{LI} when V_{CC}=0V and V_{CCO} is disconnected.
8. I_{CCO}=10 µA

DS5000FP CMOS MICROCONTROLLER



DIM	MILLIMETERS		
	MIN	NOM	MAX
A	–	2.91	3.15
B	0.25	0.35	0.45
C	–	0.80	–
D	19.85	20.00	20.15
E	13.85	14.00	14.15
F	17.40	17.86	18.20
G	23.40	23.86	24.20
H	0.40	–	1.3
I	00	50	100

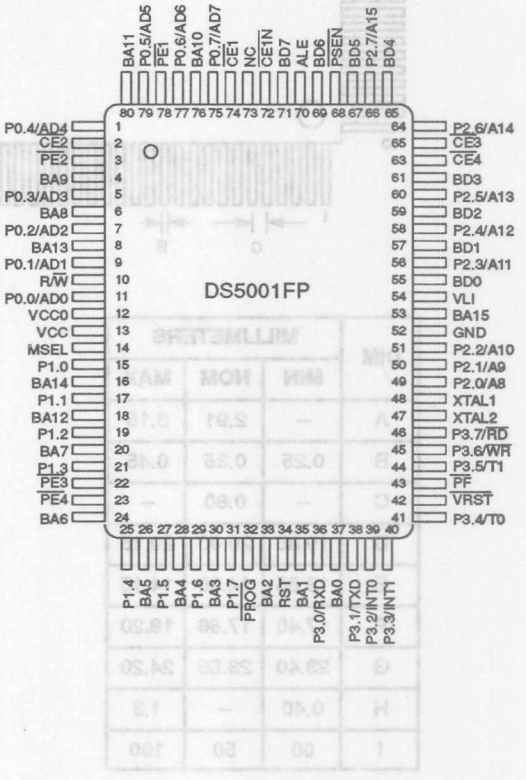
DALLAS SEMICONDUCTOR

DS5001FP
128K Soft Micro Chip

FEATURES

- 8051 compatible uC adapts to its task
  - Accesses up to 128K bytes of nonvolatile SRAM
  - In–system programming via on–chip serial port
  - Can modify its own program or data memory
  - Accesses memory on a separate Byte–wide bus
  - Performs CRC–16 check of NVRAM memory
  - Decodes memory and peripheral chip enables
- Crashproof Operation
  - Maintains all nonvolatile resources for over 10 years
  - Power–fail Reset
  - Early Warning Power–fail Interrupt
  - Watchdog Timer
  - Lithium backs user SRAM for program/data storage
  - Precision band–gap reference for power monitor
- Fully 8051 Compatible
  - 128 bytes scratchpad RAM
  - Two timer/counters
  - On–chip serial port
  - 32 parallel I/O port pins
- Software Security Available (DS5002FP)

PIN ASSIGNMENT



DESCRIPTION

The DS5001FP is an 8051 compatible microcontroller based on nonvolatile RAM technology. It is designed for systems that need large quantities of nonvolatile memory. Like its predecessor the DS5000, the DS5001FP is substantially more flexible than a standard 8051. It provides full compatibility with the 8051 instruction set, timers, serial port, and parallel I/O ports. By using NVRAM instead of ROM, the user can program, then reprogram the microcontroller while in–system. The application software can even change its own operation. This allows frequent software upgrades, adaptive programs, customized systems, etc. In addition, by using NVSRAM, the DS5001FP is ideal for data logging applications. It also connects easily to a Dallas Real–time Clock for time stamp and date.

The DS5001FP provides the benefits of NVRAM without using I/O resources. It uses a non–multiplexed Byte–wide address and data bus for memory access. This bus can perform all memory access and provides decoded chip enables for SRAM. This leaves the 32 I/O port pins free for application use. The DS5001FP uses ordinary SRAM and battery backs the memory contents with an user's external lithium cell. Data is maintained for over 10 years with a very small lithium cell. A DS5001FP also provides crashproof operation in portable systems or systems with unreliable power. These features include the ability to save the operating state, Power–fail Reset, Power–fail Interrupt, and Watchdog Timer.

A user loads programs into the DS5001FP via its on–chip Serial Bootstrap Loader. This function supervises the loading of software into NVRAM, validates it, then becomes transparent to the user. Software can be stored in multiple 32K or one 128K byte CMOS SRAM(s). Using its internal Partitioning, the DS5001FP can divide a common RAM into user selectable program and data segments. This Partition can be selected at program loading time, but can be modified anytime later. The micro will decode memory access to the SRAM, access memory via its Byte–wide bus and write–protect the memory portion designated as ROM. Combining program and data storage in one device saves board space and cost.

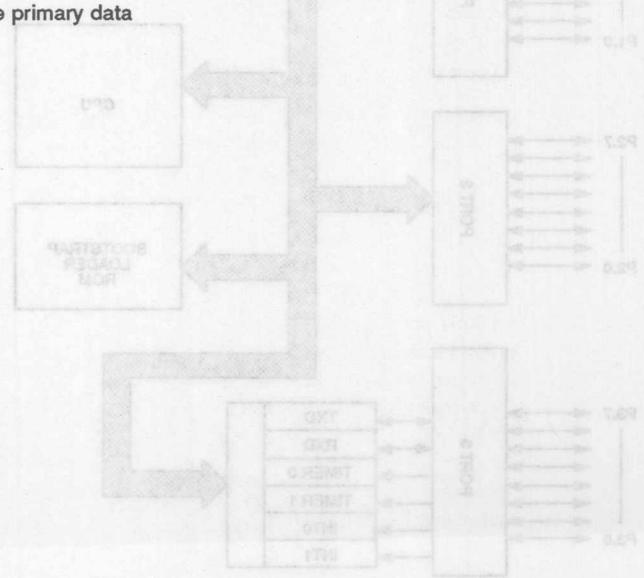
The DS5001FP offers several bank switches for access to even more memory. In addition to the primary data area of 64K bytes, a peripheral selector creates a second 64K byte data space with four accompanying chip enables. This area can be used for memory mapped peripherals or more data storage. The DS5001FP can also use its Expanded bus on Ports 0 and 2 (like an 8051) to access an additional 64K bytes of data space. Lastly, the DS5001FP provides one additional bank switch that changes up to 60K bytes of the NVRAM program space into data memory. Thus with a small amount of logic, the DS5001 accesses up to 252K bytes of data memory.

For a user that wants a pre–constructed module using the DS5001FP, RAM, lithium cell, and optional clock; the DS2251(T) is available and described in separate data sheet. More details are also contained in the User's Guide section of the Soft Microcontroller Data Book. For users that desire software security, the DS5002FP is functionally identical to the DS5001FP but provides the best firmware security available.

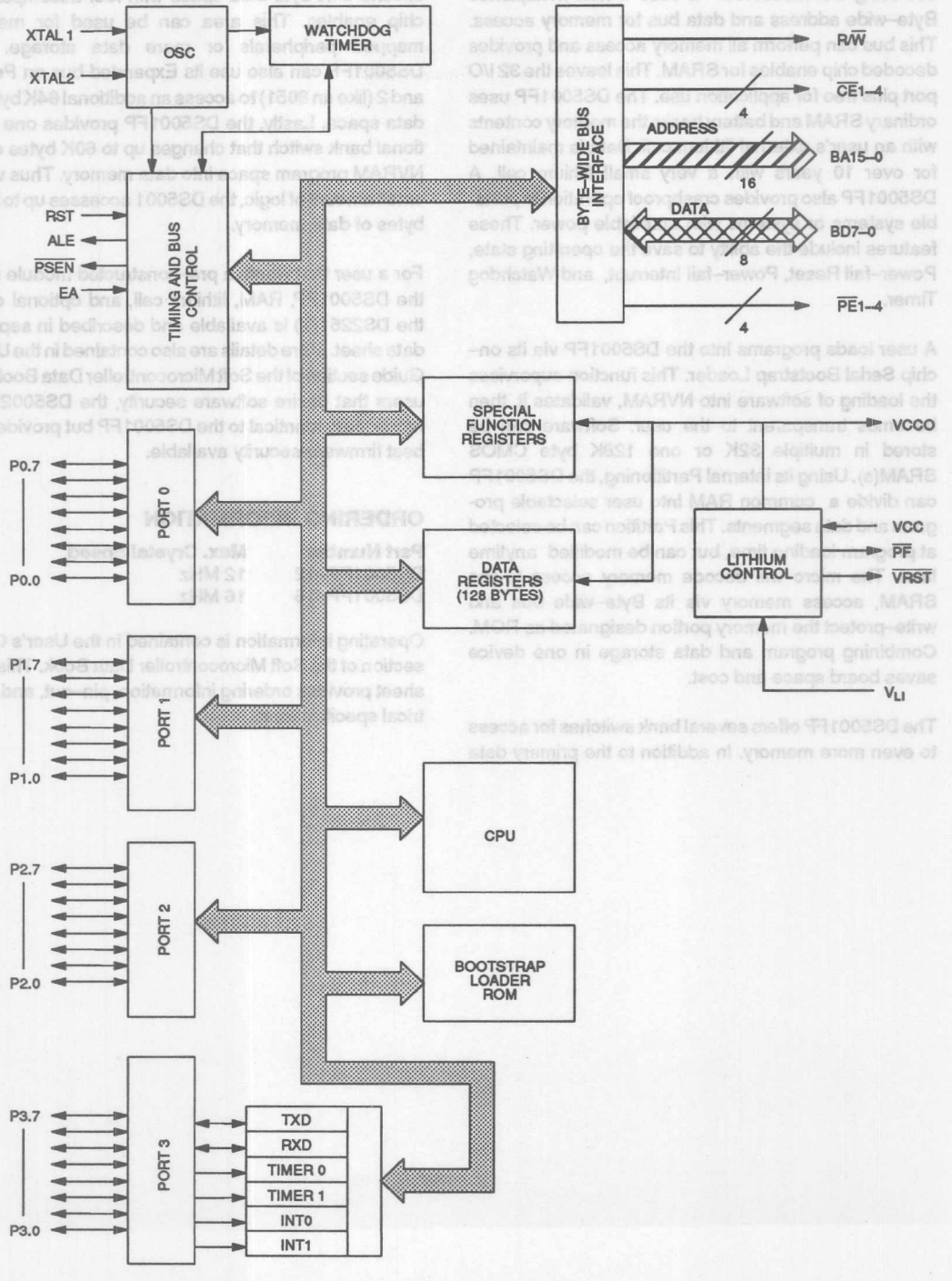
ORDERING INFORMATION

Part Number	Max. Crystal Speed
DS5001FP–12	12 MHz
DS5001FP–16	16 MHz

Operating information is contained in the User's Guide section of the Soft Microcontroller Data Book. This data sheet provides ordering information, pin–out, and electrical specifications.



DS5001FP BLOCK DIAGRAM Figure 1



PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
11, 9, 7, 5, 1, 79, 77, 75	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull–ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
15, 17, 19, 21, 25, 27, 29, 31	P1.0–P1.7 General purpose I/O Port 1.
49, 50, 51, 56, 58, 60, 64, 66	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the address in expanded memory accesses, and as pins of the RPC mode when used.
36	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
38	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
39	P3.2 $\overline{\text{INT0}}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
40	P3.3 $\overline{\text{INT1}}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
41	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
44	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
45	P3.6 $\overline{\text{WR}}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
46	P3.7 $\overline{\text{RD}}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
34	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally so this pin can be left unconnected if not used. An RC power–on reset circuit is not needed and is NOTrecommended.
68	$\overline{\text{PSEN}}$ Program Store Enable. This active low signal is used to enable an external program memory when using the Expanded bus. It is normally an output and should be unconnected if not used. $\overline{\text{PSEN}}$ also is used to invoke the Bootstrap Loader. At this time, $\overline{\text{PSEN}}$ will be pulled down externally. This should only be done once the DS5001FP is already in a reset state. The device that pulls down should be open drain since it must not interfere with $\overline{\text{PSEN}}$ under normal operation.
70	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch.

PIN NUMBER	DESCRIPTION
47, 48	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
52	GND Logic ground.
13	V_{CC} +5V
12	V_{CCO} V_{CC} Output. This is switched between V_{CC} and V_{LI} by internal circuits based on the level of V_{CC}. When power is above the lithium input, power will be drawn from V_{CC}. The lithium cell remains isolated from a load. When V_{CC} is below V_{LI}, the V_{CCO} switches to the V_{LI} source. V_{CCO} should be connected to the V_{CC} pin of an SRAM.
54	V_{LI} Lithium Voltage Input. Connect to a lithium cell greater than V_{LImin} and no greater than V_{LImax} as shown in the electrical specifications. Nominal value is +3V.
53, 16, 8, 18, 80, 76, 4, 6, 20, 24, 26, 28, 30, 33, 35, 37	BA15–0 Byte–wide Address bus bits 15–0. This bus is combined with the non–multiplexed data bus (BD7–0) to access NVSRAM. Decoding is performed using $\overline{CE1}$ through $\overline{CE4}$. Therefore, BA15 is not actually needed except for monitoring and debugging. Read/write access is controlled by R/$\overline{W}$. BA14–0 connect directly to an 8K, 32K, or 128K SRAM. If an 8K RAM is used, BA13 and BA14 will be unconnected. If a 128K SRAM is used, the micro converts $\overline{CE2}$ and $\overline{CE3}$ to serve as A16 and A15 respectively.
71, 69, 67, 65, 61, 59, 57, 55	BD7–0 Byte–wide Data bus bits 7–0. This 8 bit bi–directional bus is combined with the non–multiplexed address bus (BA14–0) to access NVSRAM. BD7–0 connect directly to an SRAM, and optionally to a Real–time Clock or other peripheral.
10	R/$\overline{W}$ Read/Write. This signal provides the write enable to the SRAMs on the Byte–wide bus. It is controlled by the memory map and Partition. The blocks selected as Program (ROM) will be write protected.
74	$\overline{CE1}$ Chip Enable 1. This is the primary decoded chip enable for memory access on the Byte–wide bus. It connects to the chip enable input of one SRAM. $\overline{CE1}$ is lithium backed. It will remain in a logic high inactive state when V_{CC} falls below V_{LI}.
72	$\overline{CE1N}$ Non battery backed version of chip enable 1. This can be used with a 32K byte EPROM. It should not be used with a battery backed chip.
2	$\overline{CE2}$ Chip Enable 2. This chip enable is provided to access a second 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, the micro converts $\overline{CE2}$ into A16 for a 128K x 8 SRAM. $\overline{CE2}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.
63	$\overline{CE3}$ Chip Enable 3. This chip enable is provided to access a third 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, the micro converts $\overline{CE3}$ into A15 for a 128K x 8 SRAM. $\overline{CE3}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.

PIN NUMBER	DESCRIPTION
62	$\overline{CE4}$ Chip Enable 4. This chip enable is provided to access a fourth 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, this signal is unused. $\overline{CE4}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.
78	$\overline{PE1}$ Peripheral Enable 1. Accesses data memory between addresses 0000h and 3FFFh when the PES bit is set to a logic 1. Commonly used to chip enable a Byte–wide real–time Clock such as the DS1283. $\overline{PE1}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}. Connect $\overline{PE1}$ to battery backed functions only.
3	$\overline{PE2}$ Peripheral Enable 2. Accesses data memory between addresses 4000h and 7FFFh when the PES bit is set to a logic 1. $\overline{PE2}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}. Connect $\overline{PE2}$ to battery backed functions only.
22	$\overline{PE3}$ Peripheral Enable 3. Accesses data memory between addresses 8000h and BFFFh when the PES bit is set to a logic 1. $\overline{PE3}$ is not lithium backed and can be connected to any type of peripheral function. If connected to a battery backed chip, it will need additional circuitry to maintain the chip enable in an inactive state when $V_{CC} < V_{LI}$.
23	$\overline{PE4}$ Peripheral Enable 4. Accesses data memory between addresses C000h and FFFFh when the PES bit is set to a logic 1. $\overline{PE4}$ is not lithium backed and can be connected to any type of peripheral function. If connected to a battery backed chip, it will need additional circuitry to maintain the chip enable in an inactivestate when $V_{CC} < V_{LI}$.
32	$\overline{PROG}$ Invokes the Bootstrap loader on a falling edge. This signal should be debounced so that only one edge is detected. If connected to ground, the micro will enter Bootstrap loading on power up. This signal is pulled up internally.
42	$\overline{VRST}$ This I/O pin indicates that the power supply (V_{CC}) has fallen below the V_{CCmin} level and the micro is in a reset state. When this occurs, the DS5001FP will drive this pin to a logic 0. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V. Because it is an I/O pin, it will also force a reset if pulled low externally. This allows multiple parts to synchronize their power–down resets.
43	$\overline{PF}$ This output goes to a logic 0 to indicate that the micro has switched to lithium backup. This corresponds to $V_{CC} < V_{LI}$. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V. The normal application of this signal is to control lithium powered current to isolate battery backed functions from non–battery backed functions.
14	MSEL Memory select. This signal controls the memory size selection. When MSEL= +5V, the DS5001FP expects to use 32K x 8 SRAMs. When MSEL = 0V, the DS5001FP expects to use a 128K x 8 SRAM. MSEL must be connected regardless of Partition, Mode, etc.
73	NC Do not connect.

INSTRUCTION SET

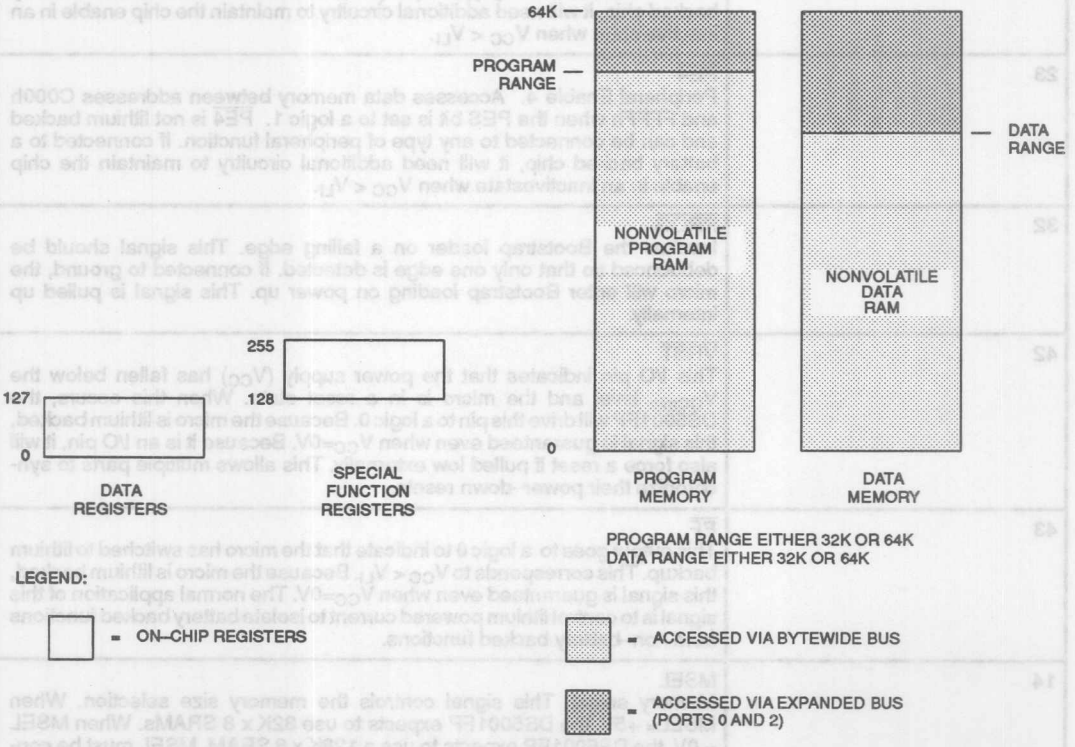
The DS5001FP executes an instruction set that is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages such as assemblers and compilers that have been written for the 8051 are compatible with the DS5001FP. A complete description of the instruction set and operation are provided in the User's Guide section of the Soft Microcontroller Data Book.

Also note that the DS5001FP is embodied in the DS2251(T) module. The DS2251(T) combines the DS5001FP with between 32K and 128K of SRAM, and a lithium cell. An optional Real–time Clock is also available in the DS2251T. This is packaged in a 72–pin SIMM module.

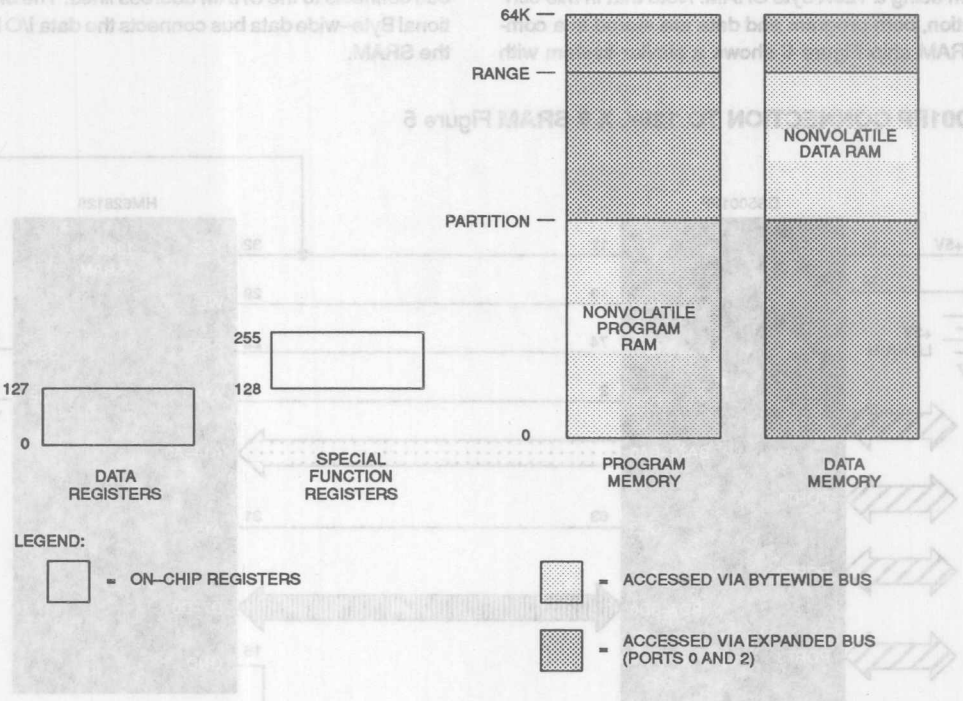
MEMORY ORGANIZATION

Figure 2 illustrates the memory map accessed by the DS5001FP. The entire 64K of program and 64K of data are potentially available to the Byte–wide bus. This preserves the I/O ports for application use. The user controls the portion of memory that is actually mapped to the Byte–wide bus by selecting the Program Range and Data Range. Any area not mapped into the NVRAM is reached via the Expanded bus on Ports 0 and 2. An alternate configuration allows dynamic Partitioning of a 64K space as shown in Figure 3. Selecting PES=1 provides another 64K of potential data storage or memory mapped peripheral space as shown in Figure 4. These selections are made using Special Function Registers. The memory map and its controls are covered in detail in the User's Guide section of the Soft Microcontroller Data Book.

MEMORY MAP OF THE DS5001FP WITH PM=1 Figure 2



MEMORY MAP OF THE DS5001FP WITH PM=0 Figure 3



MEMORY MAP OF THE DS5001FP WITH PES=1 Figure 4

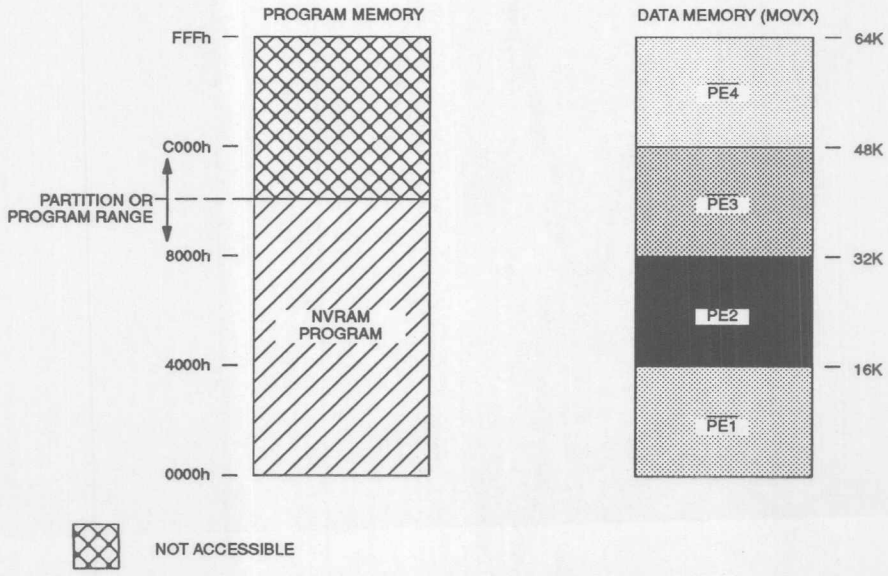
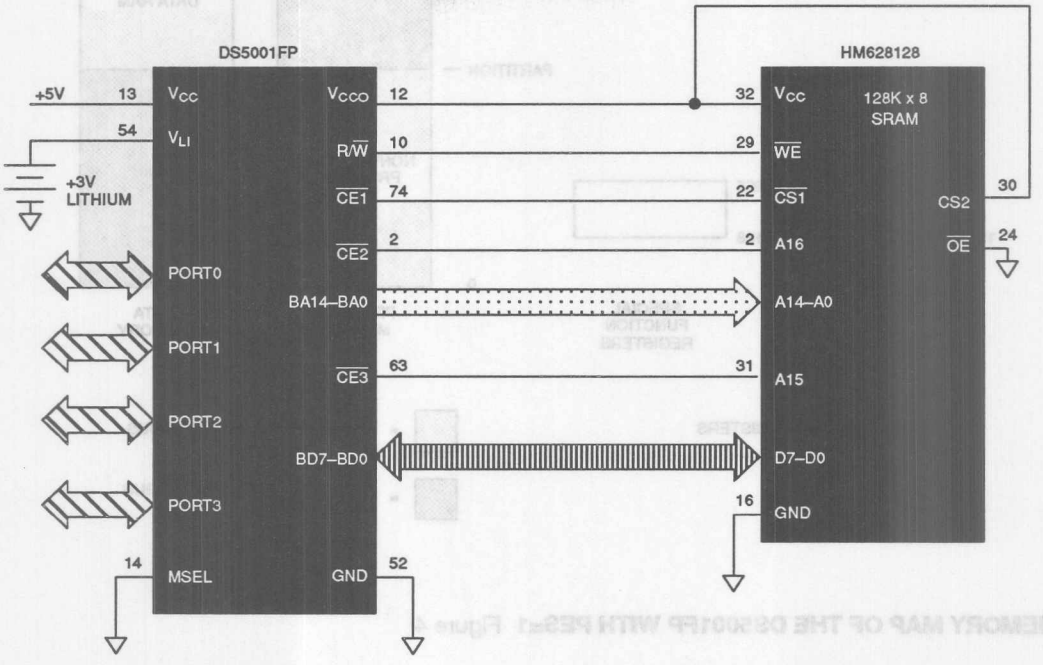
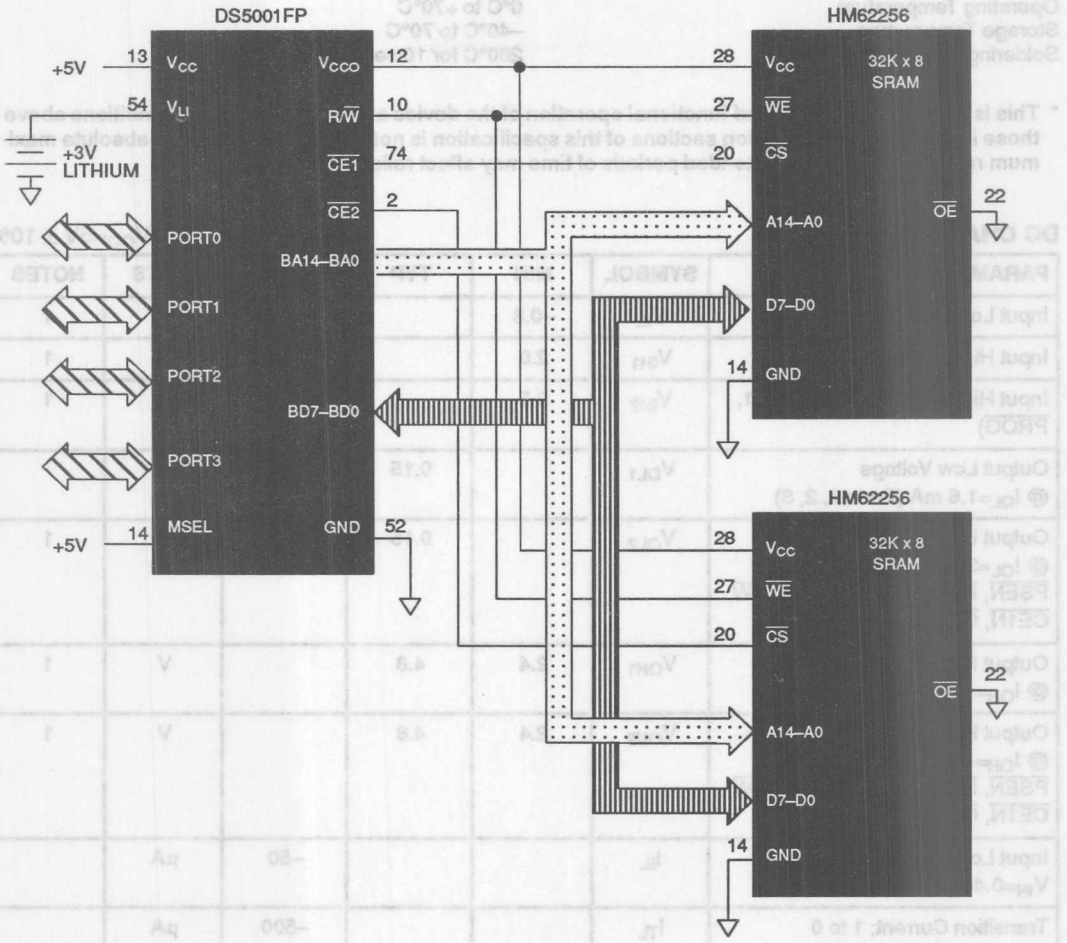


Figure 5 illustrates a typical memory connection for a system using a 128K byte SRAM. Note that in this configuration, both program and data are stored in a common RAM chip Figure 6 shows a similar system with using two 32K byte SRAMs. The Byte–wide Address bus connects to the SRAM address lines. The bi–directional Byte–wide data bus connects the data I/O lines of the SRAM.

DS5001FP CONNECTION TO 128K X 8 SRAM Figure 5



DS5001FP CONNECTION TO 64K X 8 SRAM Figure 6



POWER MANAGEMENT

The DS5001FP monitors V_{CC} to provide Power–fail Reset, early warning Power–fail Interrupt, and switch over to lithium backup. It uses an internal band–gap reference in determining the switch points. These are called V_{PFW}, V_{CCMIN}, and V_{LI} respectively. When V_{CC} drops below V_{PFW}, the DS5001FP will perform an interrupt vector to location 2Bh if the power fail warning was enabled. Full processor operation continues regardless. When power falls further to V_{CCMIN}, the DS5001FP invokes a reset state. No further code execution will be performed unless power rises back above V_{CCMIN}. All decoded chip enables and the R/$\overline{W}$ signal go to an inactive (logic 1) state. V_{CC} is still the power source at this time. When V_{CC} drops further to below V_{LI}, internal circuitry will switch to the lithium cell for power. The majority of internal circuits will be disabled and the remaining nonvolatile states will be retained. Any devices connected to V_{CCO} will be powered by the lithium cell at this time. V_{CCO} will be at the lithium battery voltage less a diode drop. This drop will vary depending on the load. Low power SRAMs should be used for this reason. When using the DS5001FP, the user must select the appropriate battery to match the RAM data retention current and the desired backup lifetime. Note that the lithium cell is only loaded when $V_{CC} < V_{LI}$. The User's Guide has more information on this topic. The trip points V_{CCMIN} and V_{PFW} are listed in the electrical specifications.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A=0°C to 70°C; V_{CC}=5V ± 10%)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	–0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC}+0.3	V	1
Input High Voltage (RST, XTAL1, $\overline{PROG}$)	V_{IH2}	3.5		V_{CC}+0.3	V	1
Output Low Voltage @ I_{OL}=1.6 mA (Ports 1, 2, 3)	V_{OL1}		0.15	0.45	V	
Output Low Voltage @ $!_{OL}$=3.2mA (Port 0, ALE, $\overline{PSEN}$, $\overline{PF}$, BA15–0, BD7–0, R/$\overline{W}$, $\overline{CE1N}$, $\overline{CE1}$–4, $\overline{PE1}$–4)	V_{OL2}		0.15	0.45	V	1
Output High Voltage @ I_{OH}=–80 mA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @ I_{OH}=–400 µA (Ports 0, ALE, $\overline{PSEN}$, $\overline{PF}$, BA15–0, BD7–0, R/$\overline{W}$, $\overline{CE1N}$, $\overline{CE1}$–4, $\overline{PE1}$–4)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN}=0.45V (Ports 1, 2, 3)	I_{IL}			–50	µA	
Transition Current; 1 to 0 V_{IN}=2.0V (Ports 1, 2, 3)	I_{TL}			–500	µA	

DC CHARACTERISTICS (cont'd) (t_A=0°C to 70°C; V_{CC}=5V ± 10%)

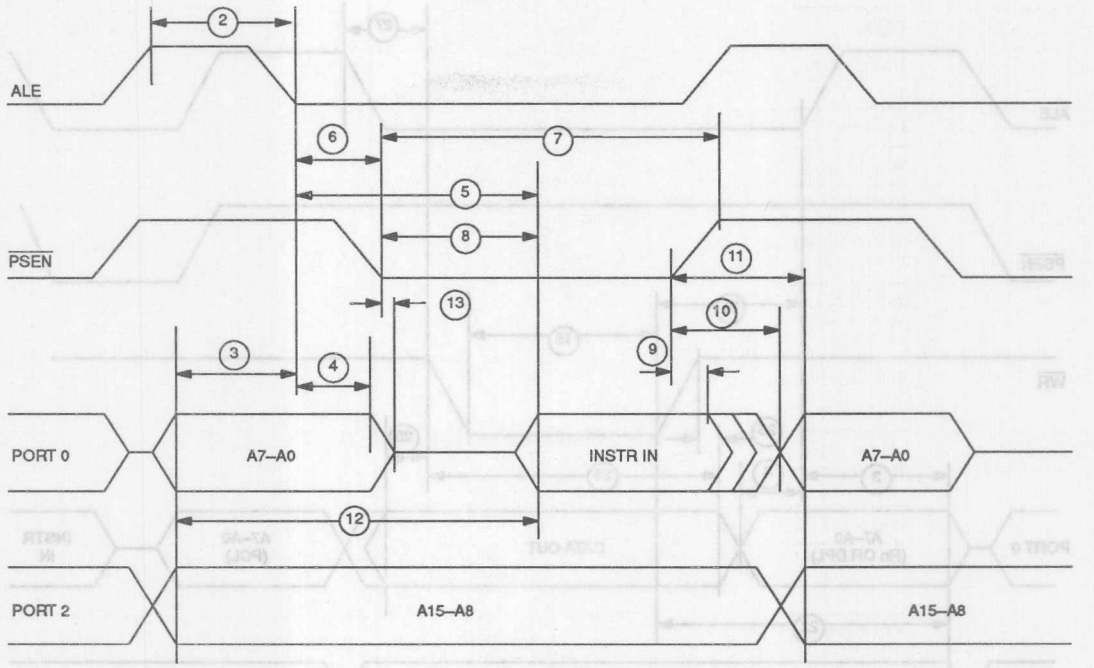
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Leakage Current 0.45<V_{IN},V_{CC} (Port 0, MSEL)	I_{IL}			±10	µA	
RST Pulldown Resistor	R_{RE}	40		150	KΩ	
$\overline{VRST}$ Pullup Resistor	R_{VR}		4.7		KΩ	
$\overline{PROG}$ Pullup Resistor	R_{PR}		40		KΩ	
Power Fail Warning Voltage	V_{PRW}	4.25	4.37	4.50	V	1
Minimum Operating Voltage	V_{CCMIN}	4.00	4.12	4.25	V	1
Lithium Supply Voltage	V_{IL}	2.5		4.0	V	1
Operating Current	I_{CC}			36	mA	2
Idle Mode Current	I_{IDLE}			7.0	mA	3
Stop Mode Current	I_{STOP}			80	µA	4
Pin Capacitance	C_{IN}			10	pF	5
Output Supply Voltage (V_{CCO})	V_{CCO1}			V_{CC}–0.3	V	1, 2
Output Supply Battery–backed Mode (V_{CCO}, $\overline{CE}$1–4, $\overline{PE}$1–2)	V_{CCO2}			V_{LI}–0.55	V	1, 8
Output Supply Current @ V_{CCO}=V_{CC} – 0.3V	I_{CCO1}			75	mA	6
Lithium–backed Quiescent Current	I_{LI}		5	75	nA	7
Reset Trip Point in Stop Mode w/BAT=3.0V w/BAT=3.3V		 4.0 4.4		 4.25 4.65		1

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

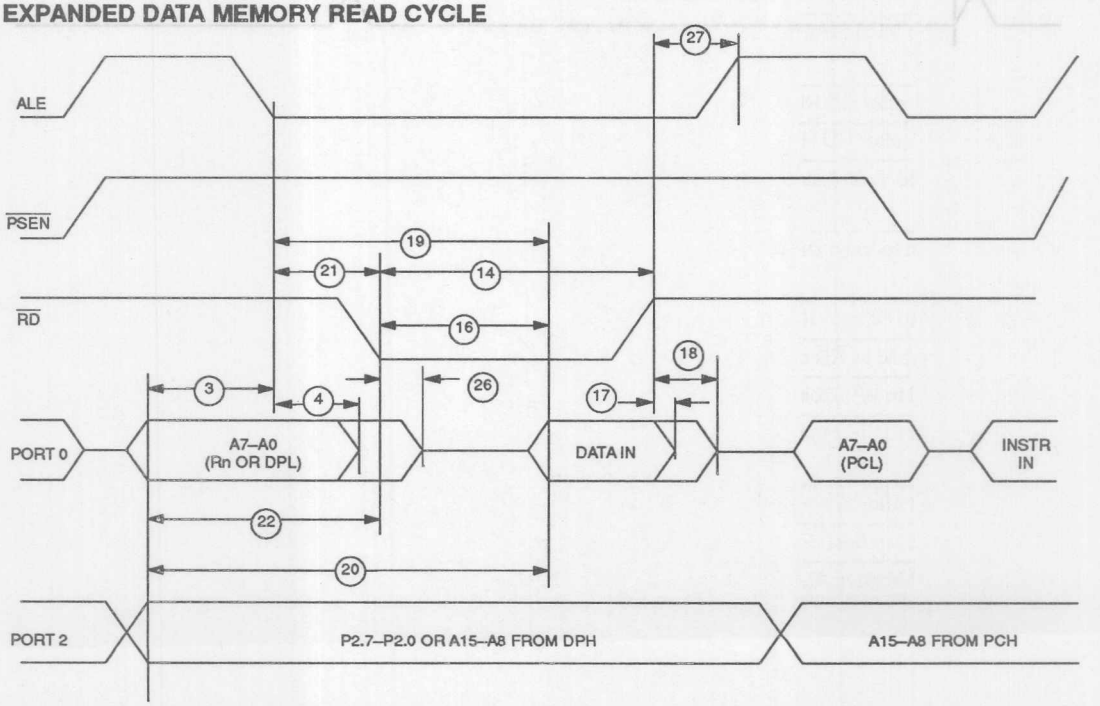
(t_A=0°C to70°C; V_{CC}=5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2t_{CLK}$–40		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK} -40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK} -35		ns
5	ALE Low to Valid Instr. In @12 MHz @16 MHz	t_{ALLVI}		$4t_{CLK}$ -150 $4t_{CLK}$ -90	ns
6	ALE Low to $\overline{PSEN}$ Low	t_{ALLPSL}	t_{CLK} -25		ns
7	$\overline{PSEN}$ Pulse Width	t_{PSPW}	$3t_{CLK}$ -35		ns
8	$\overline{PSEN}$ Low to Valid Instr. In @12 MHz @16 MHz	t_{PSLVI}		$3t_{CLK}$ -150 $3t_{CLK}$ -90	ns ns
9	Input Instr. Hold after $\overline{PSEN}$ Going High	t_{PSIV}	0		ns
10	Input Instr. Float after $\overline{PSEN}$ Going High	t_{PSIX}		t_{CLK} -20	ns
11	Address Hold after $\overline{PSEN}$ Going High	t_{PSAV}	t_{CLK} -8		ns
12	Address Valid to Valid Instr. In @12 MHz @16 MHz	t_{AVVI}		$5t_{CLK}$ -150 $5t_{CLK}$ -90	ns ns
13	$\overline{PSEN}$ Low to Address Float	t_{PSLAZ}	0		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$ -100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$ -100		ns
16	$\overline{RD}$ Low to Valid Data In @12 MHz @16 MHz	t_{RDLDV}		$5t_{CLK}$ -165 $5t_{CLK}$ -105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$ -70	ns
19	ALE Low to Valid Data In @12 MHz @16 MHz	t_{ALLVD}		$8t_{CLK}$ -150 $8t_{CLK}$ -90	ns ns
20	Valid Addr. to Valid Data In @12 MHz @16 MHz	t_{AVDV}		$9t_{CLK}$ -165 $9t_{CLK}$ -105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$ -50	$3t_{CLK}$ +50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$ -130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK} -60		ns
24	Data Valid to $\overline{WR}$ High @12 MHz @16 MHz	t_{DVWRH}	$7t_{CLK}$ -150 $7t_{CLK}$ -90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK} -50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK} -40	t_{CLK} +50	ns

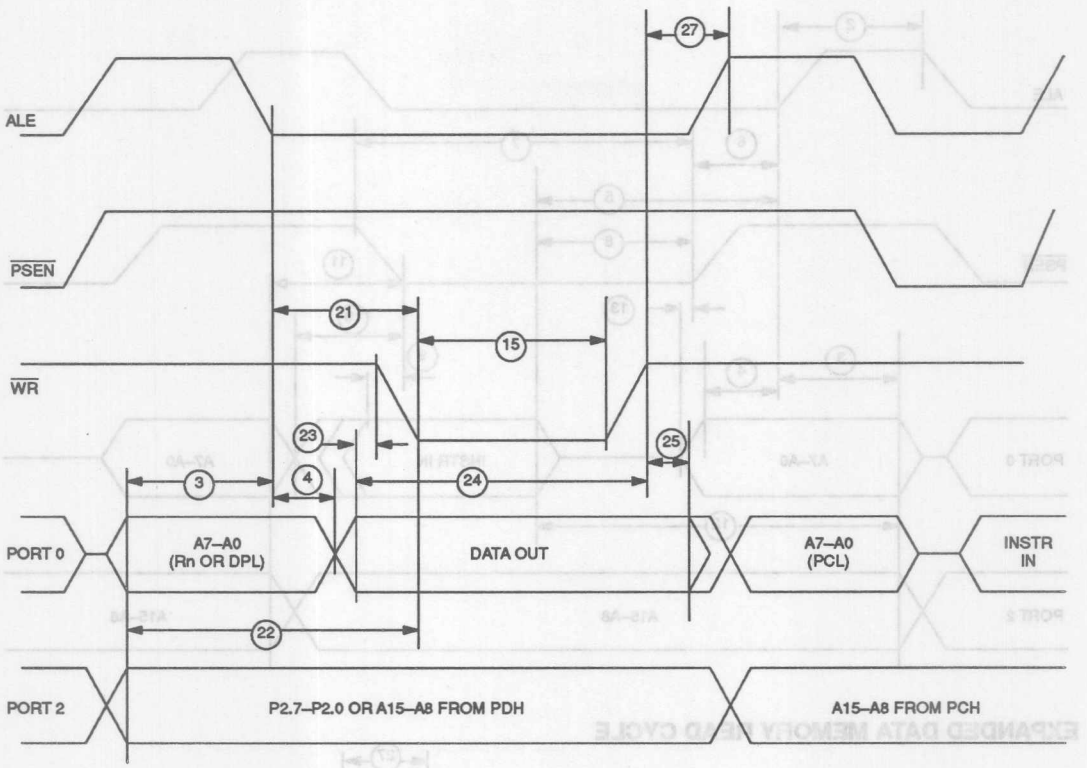
EXPANDED PROGRAM MEMORY READ CYCLE



EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE

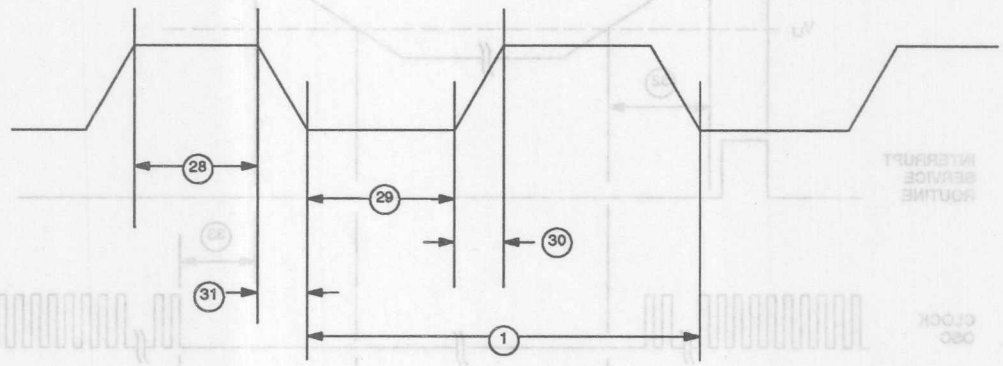


AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @16 MHz	t_{CLKF}		20 15	ns ns

EXTERNAL CLOCK TIMING

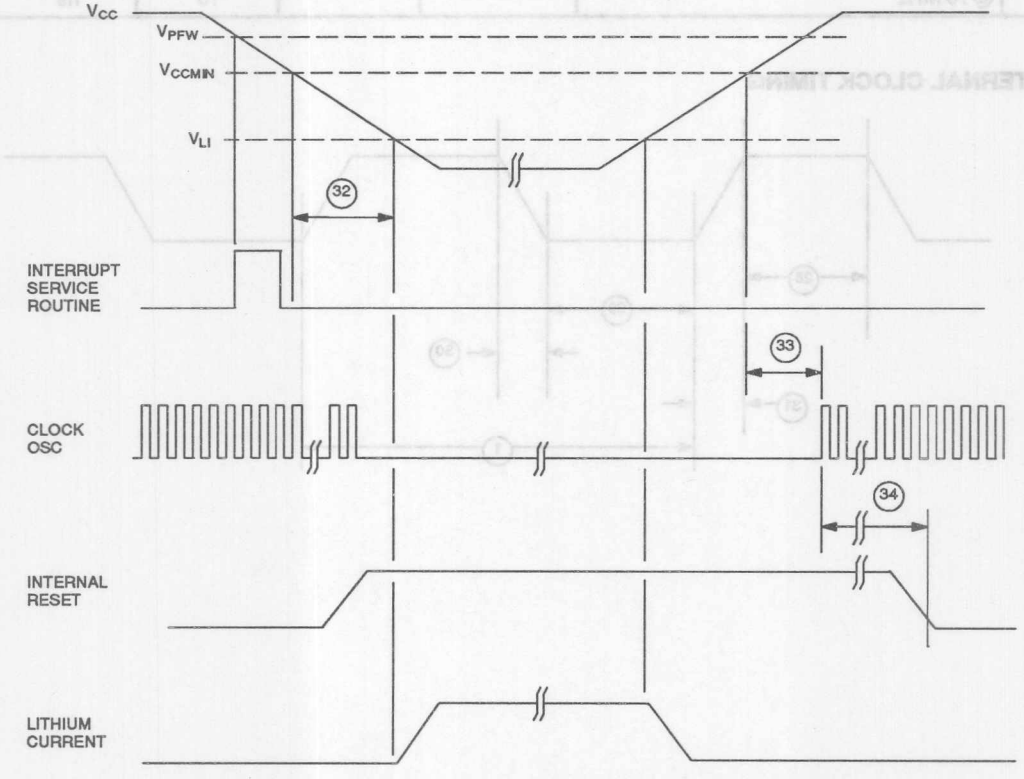


AC CHARACTERISTICS (cont'd)
POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCmin} to V_{LI}	t_F	130		µs
33	Crystal Start up Time	t_{CSU}		(note 9)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

POWER CYCLE TIMING

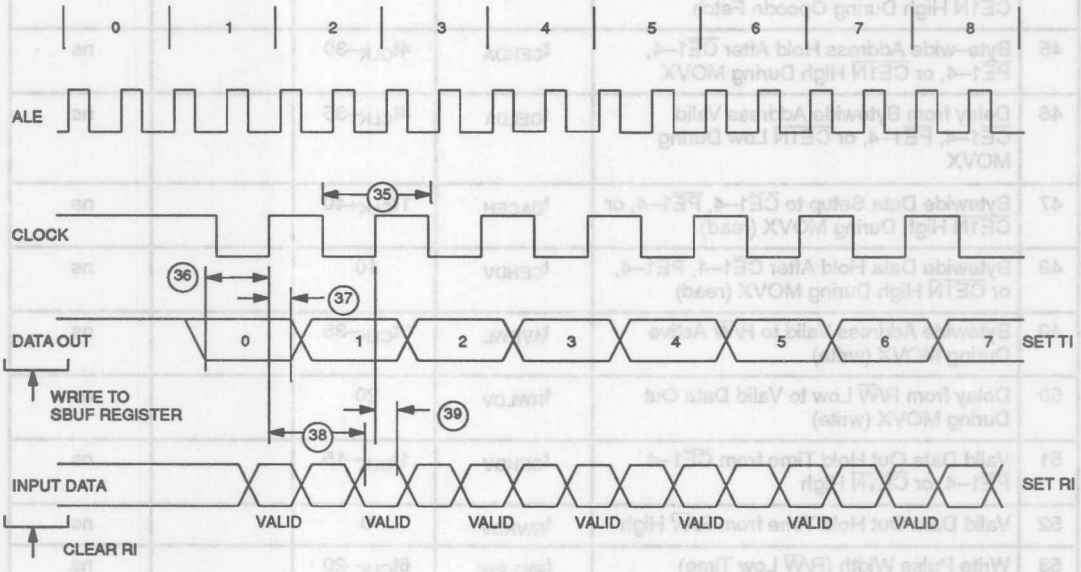


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Clock Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$ -133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$ -117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$ -133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

SERIAL PORT TIMING – MODE 0

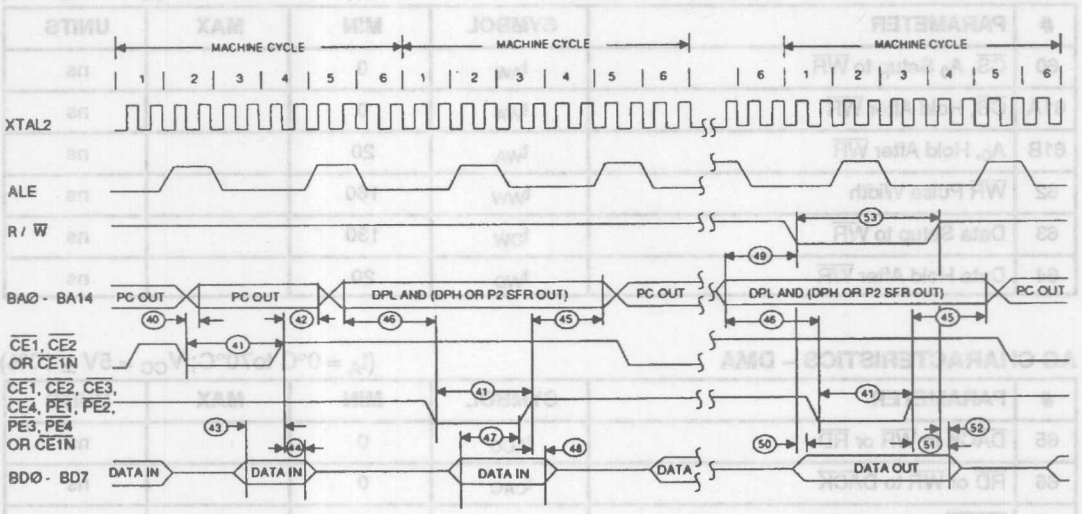


AC CHARACTERISTICS
BYTEWIDE ADDRESS/DATA BUS TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Delay to Byte–wide Address Valid from $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ Low During Opcode Fetch	t_{CE1LPA}		30	ns
41	Pulse Width of $\overline{CE1}$–4, $\overline{PE1}$–4 or $\overline{CE1N}$	t_{CEPW}	$4t_{CLK}$–35		ns
42	Byte–wide Address Hold After $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HPA}	$2t_{CLK}$–20		ns
43	Byte–wide Data Setup to $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{OVCE1H}	$1t_{CLK}$+40		ns
44	Byte–wide Data Hold After $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HOV}	10		ns
45	Byte–wide Address Hold After $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX	t_{CEHDA}	$4t_{CLK}$–30		ns
46	Delay from Bytewide Address Valid $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ Low During MOVX	t_{CELDA}	$4t_{CLK}$–35		ns
47	Bytewide Data Setup to $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{DACEH}	$1t_{CLK}$+40		ns
48	Bytewide Data Hold After $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{CEHDV}	10		ns
49	Bytewide Address Valid to R/$\overline{W}$ Active During MOVX (write)	t_{AVRWL}	$3t_{CLK}$–35		ns
50	Delay from R/$\overline{W}$ Low to Valid Data Out During MOVX (write)	t_{RWLDV}	20		ns
51	Valid Data Out Hold Time from $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High	t_{CEHDV}	$1t_{CLK}$–15		ns
52	Valid Data Out Hold Time from R/$\overline{W}$ High	t_{RWHDV}	0		ns
53	Write Pulse Width (R/$\overline{W}$ Low Time)	t_{RWLPW}	$6t_{CLK}$–20		ns

BYTEWIDE BUS TIMING



RPC AC CHARACTERISTICS – DBB READ

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
54	$\overline{CS}$, A_0 Setup to $\overline{RD}$	t_{AR}	0		ns
55	$\overline{CS}$, A_0 Hold After $\overline{RD}$	t_{RA}	0		ns
56	$\overline{RD}$ Pulse Width	t_{RR}	160		ns
57	$\overline{CS}$, A_0 to Data Out Delay	t_{AD}		130	ns
58	$\overline{RD}$ to Data Out Delay	t_{RD}	0	130	ns
59	$\overline{RD}$ to Data Float Delay	t_{RDZ}		85	ns

RPC AC CHARACTERISTICS – DBB WRITE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
60	$\overline{CS}$, A_0 Setup to $\overline{WR}$	t_{AW}	0		ns
61A	$\overline{CS}$, Hold After $\overline{WR}$	t_{WA}	0		ns
61B	A_0, Hold After $\overline{WR}$	t_{WA}	20		ns
62	$\overline{WR}$ Pulse Width	t_{WW}	160		ns
63	Data Setup to $\overline{WR}$	t_{DW}	130		ns
64	Data Hold After $\overline{WR}$	t_{WD}	20		ns

AC CHARACTERISTICS – DMA

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

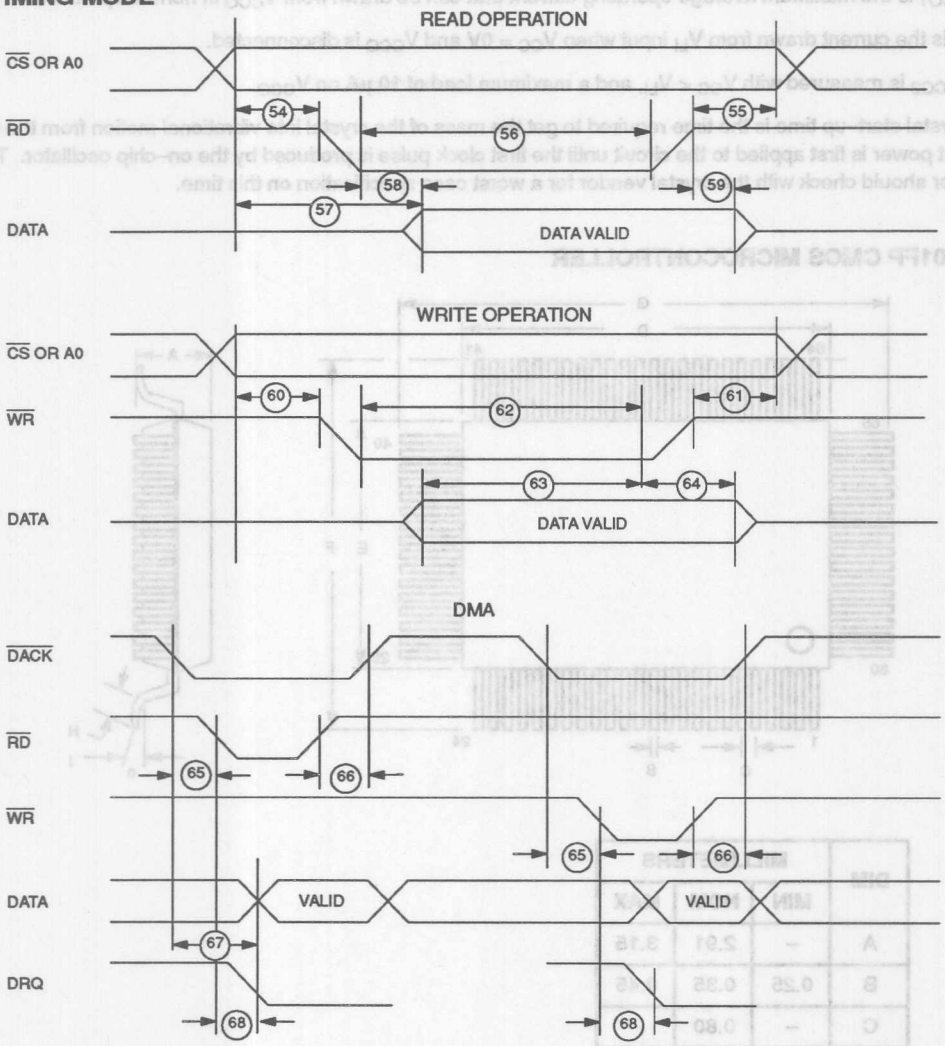
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
65	$\overline{DACK}$ to $\overline{WR}$ or $\overline{RD}$	t_{ACC}	0		ns
66	$\overline{RD}$ or $\overline{WR}$ to $\overline{DACK}$	t_{CAC}	0		ns
67	$\overline{DACK}$ to Data Valid	t_{ACD}	0	130	ns
68	$\overline{RD}$ or $\overline{WR}$ to DRQ Cleared	t_{CRQ}		110	ns

AC CHARACTERISTICS – $\overline{PROG}$

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
69	$\overline{PROG}$ Low to Active	t_{PRA}	48		CLKS
70	$\overline{PROG}$ High to Inactive	t_{PRI}	48		CLKS

RPC TIMING MODE

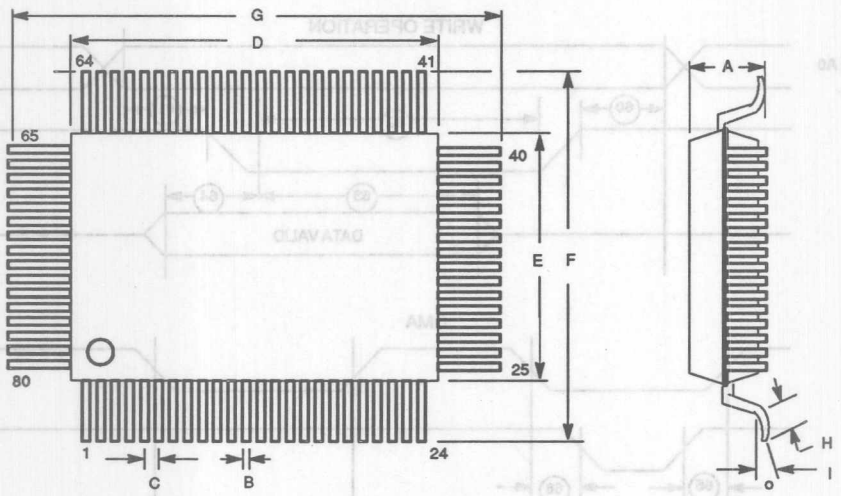


NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; RST = PORT0 = V_{CC}, MSEL = V_{SS}.
3. Idle mode I_{IDLE} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; PORT0 = V_{CC}, RST = MSEL = V_{SS}.
4. Stop mode I_{STOP} is measured with all output pins disconnected; PORT0 = V_{CC}; XTAL2 not connected; RST = MSEL = XTAL1 = V_{SS}.
5. Pin Capacitance is measured with a test frequency – 1 MHz, t_A = 25°C.

6. I_{CCO1} is the maximum average operating current that can be drawn from V_{CCO} in normal operation.
7. I_{LI} is the current drawn from V_{LI} input when V_{CC} = 0V and V_{CCO} is disconnected.
8. V_{CCO2} is measured with $V_{CC} < V_{LI}$, and a maximum load of 10 µA on V_{CCO}.
9. Crystal start–up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on–chip oscillator. The user should check with the crystal vendor for a worst case specification on this time.

DS5001FP CMOS MICROCONTROLLER



DIM	MILLIMETERS		
	MIN	NOM	MAX
A	–	2.91	3.15
B	0.25	0.35	0.45
C	–	0.80	–
D	19.85	20.00	20.15
E	13.85	14.00	14.15
F	17.40	17.86	18.20
G	23.40	23.86	24.20
H	0.40	–	1.3
I	0	–	10°

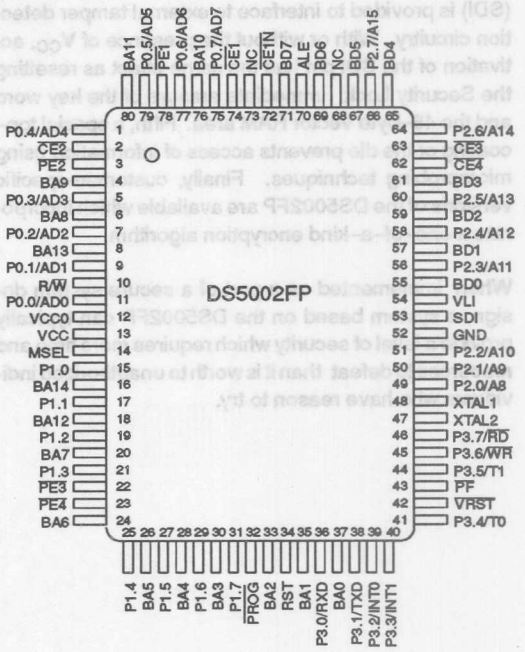
DALLAS SEMICONDUCTOR

DS5002FP
Secure Micro

FEATURES

- 8051 compatible uC for secure/sensitive applications
  - Access 32, 64, or 128K bytes of nonvolatile SRAM for program and/or data storage
  - In–system programming via on–chip serial port
  - Capable of modifying its own program or data memory in the end system
- Firmware Security Features:
  - Memory stored in encrypted form
  - Encryption using on–chip 64–bit key
  - Automatic true random key generator
  - SDI Self Destruct Input
  - Top coating prevent microprobe
  - Improved security over previous generations
  - Protects memory contents from piracy
- Crashproof Operation
  - Maintains all nonvolatile resources for over 10 years in the absence of power
  - Power–fail Reset
  - Early Warning Power–fail Interrupt
  - Watchdog Timer

PIN ASSIGNMENT



DESCRIPTION

The DS5002FP Secure Micro Chip is a secure version of the DS5001FP 128K Micro Chip. In addition to the memory and I/O enhancements of the DS5001FP, the Secure Micro Chip incorporates the most sophisticated security features available in any microcontroller. The security features of the DS5002FP include an array of mechanisms which are designed to resist all levels of threat, including observation, analysis, and physical attack. As a result, a massive effort would be required to obtain any information about memory contents. Furthermore, the "soft" nature of the DS5002FP allows frequent modification of the secure information, thereby minimizing the value of any secure information obtained by such a massive effort.

The DS5002FP implements a security system which is an improved version of its predecessor, the DS5000. Like the DS5000, the DS5002FP loads and executes application software in encrypted form. Up to 128K x 8 bytes of standard SRAM can be accessed via its byte–wide bus. This RAM is converted by the DS5002FP into lithium–backed nonvolatile storage for program and data. As a result, the contents of the RAM and the execution of the software appear unintelligible to the outside observer. The encryption algorithm uses an internally stored and protected key. Any attempt to discover the key value results in its erasure, rendering the encrypted contents of the RAM useless.

The Secure Micro Chip offers a number of major enhancements to the software security implemented in the previous generation DS5000. First, the DS5002FP provides a stronger software encryption algorithm which incorporates elements of DES encryption. Second, the encryption is based on a 64–bit key word, as compared to the DS5000's 40–bit key. Third, the key can only be loaded from an on–chip true random number generator. As a result, the true key value is never known by the user. Fourth, a Self–Destruct input pin (SDI) is provided to interface to external tamper detection circuitry. With or without the presence of V_{CC}, activation of the SDI pin has the same effect as resetting the Security Lock: immediate erasure of the key word and the 48–byte Vector RAM area. Fifth, a special top–coating of the die prevents access of information using microprobing techniques. Finally, customer–specific versions of the DS5002FP are available which incorporate a one–of–a–kind encryption algorithm.

When implemented as a part of a secure system design, a system based on the DS5002FP can typically provide a level of security which requires more time and resources to defeat than it is worth to unauthorized individuals who have reason to try.

ORDERING INFORMATION

The following versions of the DS5002FP are available as standard products from Dallas Semiconductor:

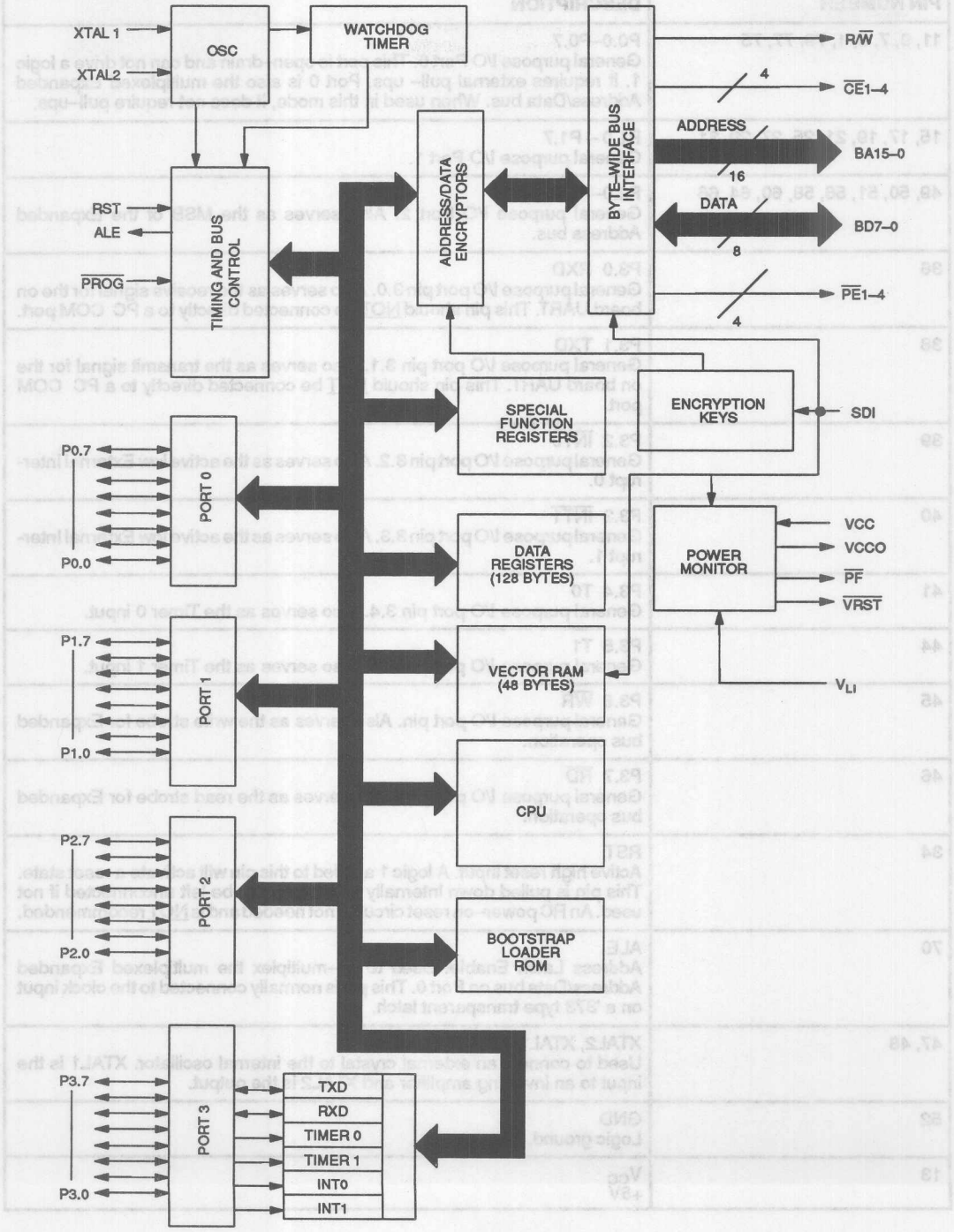
PART #	CLOCK	PACKAGE
DS5002FP–12	12 MHz	80–pin QFP
DS5002FP–16	16 MHz	80–pin QFP
DS5002FPM–12	12 MHz	80–pin QFP with metal
DS5002FPM–16	16 MHz	80–pin QFP with metal

Please contact Dallas Semiconductor for ordering information on customer–specific versions of the DS5002FP.

BLOCK DIAGRAM

Figure 1 is a block diagram illustrating the internal architecture of the DS5002. The DS5002 is a secure implementation of the DS5001 128K Micro Chip. As a result, it operates in an identical fashion to the DS5001 except where indicated. See the DS5001 Data Sheet for operating details.

DS5002 BLOCK DIAGRAM Figure 1



PIN DESCRIPTION

PIN NUMBER	DESCRIPTION
11, 9, 7, 5, 1, 79, 77, 75	P0.0–P0.7 General purpose I/O Port 0. This port is open–drain and can not drive a logic 1. It requires external pull– ups. Port 0 is also the multiplexed Expanded Address/Data bus. When used in this mode, it does not require pull–ups.
15, 17, 19, 21, 25, 27, 29, 31	P1.0 – P1.7 General purpose I/O Port 1.
49, 50, 51, 56, 58, 60, 64, 66	P2.0–P2.7 General purpose I/O Port 2. Also serves as the MSB of the Expanded Address bus.
36	P3.0 RXD General purpose I/O port pin 3.0. Also serves as the receive signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
38	P3.1 TXD General purpose I/O port pin 3.1. Also serves as the transmit signal for the on board UART. This pin should NOT be connected directly to a PC COM port.
39	P3.2 $\overline{INT0}$ General purpose I/O port pin 3.2. Also serves as the active low External Interrupt 0.
40	P3.3 $\overline{INT1}$ General purpose I/O port pin 3.3. Also serves as the active low External Interrupt 1.
41	P3.4 T0 General purpose I/O port pin 3.4. Also serves as the Timer 0 input.
44	P3.5 T1 General purpose I/O port pin 3.5. Also serves as the Timer 1 input.
45	P3.6 $\overline{WR}$ General purpose I/O port pin. Also serves as the write strobe for Expanded bus operation.
46	P3.7 $\overline{RD}$ General purpose I/O port pin. Also serves as the read strobe for Expanded bus operation.
34	RST Active high reset input. A logic 1 applied to this pin will activate a reset state. This pin is pulled down internally so this pin can be left unconnected if not used. An RC power–on reset circuit is not needed and is NOT recommended.
70	ALE Address Latch Enable. Used to de–multiplex the multiplexed Expanded Address/Data bus on Port 0. This pin is normally connected to the clock input on a '373 type transparent latch.
47, 48	XTAL2, XTAL1 Used to connect an external crystal to the internal oscillator. XTAL1 is the input to an inverting amplifier and XTAL2 is the output.
52	GND Logic ground.
13	V_{CC} +5V

PIN NUMBER	DESCRIPTION
12	V_{CCO} V_{CC} Output. This is switched between V_{CC} and V_{LI} by internal circuits based on the level of V_{CC}. When power is above the lithium input, power will be drawn from V_{CC}. The lithium cell remains isolated from a load. When V_{CC} is below V_{LI}, the V_{CCO} switches to the V_{LI} source. V_{CCO} should be connected to the V_{CC} pin of an SRAM.
54	V_{LI} Lithium Voltage Input. Connect to a lithium cell greater than V_{LImin} and no greater than V_{LImax} as shown in the electrical specifications. Nominal value is +3V.
16, 8, 18, 80, 76, 4, 6, 20, 24, 26, 28, 30, 33, 35, 37	BA14–0 Byte–wide Address bus bits 14–0. This bus is combined with the non–multiplexed data bus (BD7–0) to access NVSRAM. Decoding is performed using $\overline{CE1}$ through $\overline{CE4}$. Therefore, BA15 is not actually needed. Read/write access is controlled by R/$\overline{W}$. BA14–0 connect directly to an 8K, 32K, or 128K SRAM. If an 8K RAM is used, BA13 and BA14 will be unconnected. If a 128K SRAM is used, the micro converts $\overline{CE2}$ and $\overline{CE3}$ to serve as A16 and A15 respectively.
71, 69, 67, 65, 61, 59, 57, 55	BD7–0 Byte–wide Data bus bits 7–0. This 8 bit bi–directional bus is combined with the non–multiplexed address bus (BA14–0) to access NVSRAM. Decoding is performed on $\overline{CE1}$ and $\overline{CE2}$. Read/write access is controlled by R/$\overline{W}$. BD7–0 connect directly to an SRAM, and optionally to a Real–time Clock or other peripheral.
10	R/$\overline{W}$ Read/Write. This signal provides the write enable to the SRAMs on the Byte–wide bus. It is controlled by the memory map and Partition. The blocks selected as Program (ROM) will be write protected.
74	$\overline{CE1}$ Chip Enable 1. This is the primary decoded chip enable for memory access on the Byte–wide bus. It connects to the chip enable input of one SRAM. $\overline{CE1}$ is lithium backed. It will remain in a logic high inactive state when V_{CC} falls below V_{LI}.
2	$\overline{CE2}$ Chip Enable 2. This chip enable is provided to access a second 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, the micro converts $\overline{CE2}$ into A16 for a 128K x 8 SRAM. $\overline{CE2}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.
63	$\overline{CE3}$ Chip Enable 3. This chip enable is provided to access a third 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, the micro converts $\overline{CE3}$ into A15 for a 128K x 8 SRAM. $\overline{CE3}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.
62	$\overline{CE4}$ Chip Enable 4. This chip enable is provided to access a fourth 32K block of memory. It connects to the chip enable input of one SRAM. When MSEL=0, this signal is unused. $\overline{CE4}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}.
78	$\overline{PE1}$ Peripheral Enable 1. Accesses data memory between addresses 0000h and 3FFFh when the PES bit is set to a logic 1. Commonly used to chip enable a Byte–wide real–time Clock such as the DS1283. $\overline{PE1}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}. Connect $\overline{PE1}$ to battery backed functions only.

PIN NUMBER	DESCRIPTION
3	$\overline{PE2}$ Peripheral Enable 2. Accesses data memory between addresses 4000h and 7FFFh when the PES bit is set to a logic 1. $\overline{PE2}$ is lithium backed and will remain at a logic high when V_{CC} falls below V_{LI}. Connect $\overline{PE2}$ to battery backed functions only.
22	$\overline{PE3}$ Peripheral Enable 3. Accesses data memory between addresses 8000h and BFFFh when the PES bit is set to a logic 1. $\overline{PE3}$ is not lithium backed and can be connected to any type of peripheral function. If connected to a battery backed chip, it will need additional circuitry to maintain the chip enable in an inactive state when $V_{CC} < V_{LI}$.
23	$\overline{PE4}$ Peripheral Enable 4. Accesses data memory between addresses C000h and FFFFh when the PES bit is set to a logic 1. $\overline{PE4}$ is not lithium backed and can be connected to any type of peripheral function. If connected to a battery backed chip, it will need additional circuitry to maintain the chip enable in an inactive state when $V_{CC} < V_{LI}$.
32	$\overline{PROG}$ Invokes the Bootstrap loader on a falling edge. This signal should be debounced so that only one edge is detected. If connected to ground, the micro will enter Bootstrap loading on power up. This signal is pulled up internally.
42	$\overline{VRST}$ This I/O pin indicates that the power supply (V_{CC}) has fallen below the V_{CCmin} level and the micro is in a reset state. When this occurs, the DS5002FP will drive this pin to a logic 0. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V. Because it is an I/O pin, it will also force a reset if pulled low externally. This allows multiple parts to synchronize their power–down resets.
43	$\overline{PF}$ This output goes to a logic 0 to indicate that the micro has switched to lithium backup. This corresponds to $V_{CC} < V_{LI}$. Because the micro is lithium backed, this signal is guaranteed even when V_{CC}=0V. The normal application of this signal is to control lithium powered current to isolate battery backed functions form non–battery backed functions.
14	MSEL Memory select. This signal controls the memory size selection. When MSEL= +5V, the DS5002FP expects to use 32K x 8 SRAMs. When MSEL = 0V, the DS5002FP expects to use a 128K x 8 SRAM. MSEL must be connected regardless of Partition, Mode, etc.
53	SDI Self–Destruct Input. An active high on this pin causes an unlock procedure. This results in the destruction of Vector RAM, Encryption Keys, and the loss of power from V_{CCO}. This pin should be grounded if not used.
72	$\overline{CE1N}$ This is a non–battery backed version of $\overline{CE1}$. It is not generally useful since the DS5002 can not be used with EPROM due to its encryption.
73	NC Do not connect.

SECURE OPERATION OVERVIEW

The DS5002FP incorporates encryption of the activity on its Byte–wide Address/Data bus to prevent unauthorized access to the program and data information contained in the nonvolatile RAM. Loading an application program in this manner is performed via the Bootstrap Loader using the general sequence described below:

1. Clear Security Lock
2. Set memory map configuration as for DS5001
3. Load application software
4. Set Security Lock
5. Exit Loader

Loading of application software into the program/data RAM is performed while the DS5002 is in its Bootstrap Load mode. Loading is only possible when the Security Lock is clear. If the Security Lock has previously set, then it must be cleared by issuing the "Z" command from the Bootstrap Loader. Resetting the security lock instantly clears the previous key word and the contents of the vector RAM. In addition, the Bootstrap ROM writes zeroes into the first 32K of external RAM.

The user's application software is loaded into external CMOS SRAM via the "L" command in "scrambled" form through on–chip encryptor circuits. Each external RAM address is an encrypted representation of an on–chip logical address. Thus, the sequential instructions of an ordinary program or data table are stored non–sequentially in RAM memory. The contents of the program/data RAM are also encrypted. Each byte in RAM is encrypted by a key– and address–dependent encryptor circuit such that identical bytes are stored as different values in different memory locations.

The encryption of the program/data RAM is dependent on an on–chip 64–bit key word. The key is loaded by the ROM firmware just prior to the time that the application software is loaded, and is retained as nonvolatile information in the absence of V_{CC} by the lithium backup circuits. After loading is complete, the key is protected by setting the on–chip Security Lock, which is also retained as nonvolatile information in the absence of V_{CC}. Any attempt to tamper with the key word and thereby gain access to the true program/data RAM contents results in the erasure of the key word as well as the RAM contents.

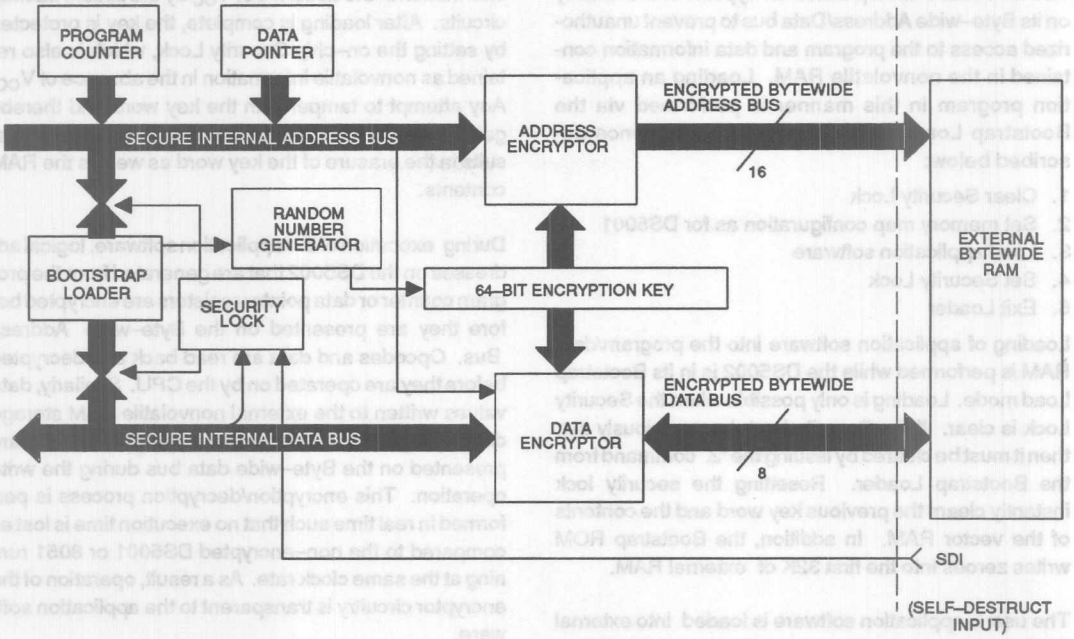
During execution of the application software, logical addresses on the DS5002 that are generated from the program counter or data pointer registers are encrypted before they are presented on the Byte–wide Address Bus. Opcodes and data are read back and decrypted before they are operated on by the CPU. Similarly, data values written to the external nonvolatile RAM storage during program execution are encrypted before they are presented on the Byte–wide data bus during the write operation. This encryption/decryption process is performed in real time such that no execution time is lost as compared to the non–encrypted DS5001 or 8051 running at the same clock rate. As a result, operation of the encryptor circuitry is transparent to the application software.

Unlike the DS5000, the DS5002FP chip's security feature is always enabled.

SECURITY CIRCUITRY

The on–chip functions associated with the DS5002's software security feature are depicted in Figure 2. Encryption logic consists of an address encryptor and a data encryptor. Although each encryptor uses its own algorithm for encrypting data, both depend on the 64–bit key word which is contained in the encryption key registers. Both the encryptors operate during loading of the application software and also during its execution.

DS5002 SECURITY CIRCUITRY Figure 2



The address encryptor translates each "logical" address, i.e., the normal sequence of addresses that are generated in the logical flow of program execution, into an encrypted address (or "physical" address) at which the byte is actually stored. Each time a logical address is generated, either during program loading or during program execution, the address encryptor circuitry uses the value of the 64–bit key word and of the address itself to form the physical address which will be presented on the address lines of the RAM. The encryption algorithm is such that there is one and only one physical address for every possible logical address. The address encryptor operates over the entire memory range which is configured during Bootstrap Loading for access on the Byte–wide Bus.

As Bootstrap Loading of the application software is performed, the Data Encryptor logic transforms the opcode, operand, or data byte at any given memory location into an encrypted representation. As each byte is read back to the CPU during program execution, the internal Data Encryptor restores it to its original value. When a byte is written to the external nonvolatile program/data RAM during program execution, that byte is stored in encrypted form as well. The data encryption logic uses the value of the 64–bit key, the logical address to which the data is being written, and the value of the data itself to form the encrypted data which is written to the nonvolatile program/data RAM. The encryption algorithm is repeatable, such that for a given data value, encryption key value, and logical address the encrypted byte will always be the same. However, there are many possible encrypted data values for each possible true data value due to the algorithm's dependency on the values of the logical address and encryption key.

When the application software is executed, the internal CPU of the DS5002 operates as normal. Logical addresses are calculated for opcode fetch cycles and also data read and write operations. The DS5002 has the ability to perform address encryption on logical addresses as they are generated internally during the normal course of program execution. In a similar fashion, data is manipulated by the CPU in its true representation. However, it is also encrypted when it is written to the external program/data RAM, and is restored to its original value when it is read back.

When an application program is stored in the format described above, it is virtually impossible to disassemble opcodes or to convert data back into its true representation. Address encryption has the effect that the op-

codes and data are not stored in the contiguous form in which they were assembled, but rather in seemingly random locations in memory. This in itself makes it virtually impossible to determine the normal flow of the program. As an added protection measure, the Address Encryptor also generates "dummy" read access cycles whenever time is available during program execution.

DUMMY READ CYCLES

Like the DS5000, the DS5002FP generates a "dummy" read access cycle to non–sequential addresses in external RAM memory whenever time is available during program execution. This action has the effect of further complicating the task of determining the normal flow of program execution. During these pseudo–random dummy cycles, the RAM is read to all appearance, but the data is not used internally. Through the use of a repeatable exchange of dummy and true read cycles, it is impossible to distinguish a dummy cycle from a real one.

ENCRYPTION ALGORITHM

The DS5002FP incorporates a proprietary algorithm implemented in hardware which performs the scrambling of address and data on the byte–wide bus to the static RAM. This algorithm has been greatly strengthened with respect t o its DS5000 predecessor. Improvements include:

1. 64–bit encryption key
2. Incorporation of DES–like operations to provide a greater degree of nonlinearity.
3. Customizable encryption

The encryption circuitry uses a 64–bit key value (compared to the DS5000's 40–bit key) which is stored on the DS5002 die and protected by the Security Lock function described below. In addition, the algorithm has been strengthened to incorporate certain operations used in DES encryption, so that the encryption of both the addresses and data is highly nonlinear. Unlike the DS5000, the encryption circuitry in the DS5002 is always enabled.

Dallas Semiconductor can customize the encryption circuitry by laser programming the die to insure that a unique encryption algorithm is delivered to the customer. In addition, the customer–specific version can be branded as specified by the customer. Please contact Dallas Semiconductor for ordering information of customer–specific versions.

ENCRYPTION KEY

As described above, the on–chip 64–bit Encryption Key is the basis of both the address and data encryptor circuits. The DS5002 provides a key management system which is greatly improved over the DS5000. The DS5002 does not give the user the ability to select a key. Instead, when the loader is given certain commands, the key is set based on the value read from an on–chip hardware random number generator. This action is performed just prior to actually loading the code into the external RAM. This scheme prevents characterization of the encryption algorithm by continuously loading new, known keys. It also frees the user from the burden of protecting the key selection process.

The random number generator circuit uses the asynchronous frequency differences of two internal ring oscillator and the processor master clock (determined by XTAL1 and XTAL2). As a result, a true random number is produced.

VECTOR RAM

A 48–byte vector RAM area is incorporated on–chip, and is used to contain the reset and interrupt vector code in the DS5002. It is included in the architecture to help insure the security of the application program.

If reset and interrupt vector locations were accessed from the external nonvolatile program/data RAM during the execution of the program, then it would be possible to determine the encrypted value of known addresses. This could be done by forcing an interrupt or reset condition and observing the resulting addresses on the Byte–wide address/data bus. For example, it is known that when a hardware reset is applied the logical program address is forced to location 0000H and code is executed starting from this location. It would then be possible to determine the encrypted value (or physical address) of the logical address value 0000H by observing the address presented to the external RAM following a hardware reset. Interrupt vector address relationships could be determined in a similar fashion. By using the on–chip vector RAM to contain the interrupt and reset vectors, it is impossible to observe such relationships. Although it is very unlikely that an application program could be deciphered by observing vector address rela-

tionships, the vector RAM eliminates this possibility. Note that the dummy accesses mentioned above are conducted while fetching from vector RAM.

The Vector RAM is automatically loaded with the user's reset and interrupt vectors during bootstrap loading.

SECURITY LOCK

Once the application program has been loaded into the DS5002's NVRAM, the Security Lock may be enabled by issuing the "Z" command in the Bootstrap Loader. While the Security Lock is set, no further access to program/ data information is possible via the on–chip ROM. Access is prevented by both the bootstrap loader firmware and the DS5002FP encryptor circuits.

Access to the NVRAM may only be regained by clearing the Security Lock via the "U" command in the Bootstrap Loader. This action triggers several events which defeat tampering. First, the encryption key is instantaneously erased. Without the encryption key, the DS5002FP is no longer able to decrypt the contents of the RAM. Therefore, the application software can no longer be correctly executed, nor can it be read back in its true form via the Bootstrap Loader. Second, the Vector RAM area is also instantaneously erased, so that the reset and vector information is lost. Third, the Bootstrap Loader firmware sequentially erases the encrypted RAM area. Lastly, the loader creates and loads a new random key.

The Security Lock bit itself is constructed using a multiple–bit latch which is interlaced for self–destruct in the event of tampering. The lock is designed to set–up a "domino–effect" such that erasure of the bit will result in an unstoppable sequence of events that clears critical data including encryption key and vector RAM. In addition, this bit is protected from probing by the top–coating feature mentioned below.

SELF–DESTRUCT INPUT

The Self–Destruct Input (SDI) pin is an active high input which is used to reset the Security Lock in response to an external event. The SDI input is intended to be used with external tamper detection circuitry. It can be activated with or without operating power applied to the V_{CC} pin. Activation of the SDI pin instantly resets the security lock and causes the same sequence of events described above for this action. In addition, power is momentarily removed from the Byte–wide bus interface including the V_{CCO} pin, resulting in the loss of data in external RAM.

TOP LAYER COATING

The DS5002 is provided with a special top–layer coating that is designed to prevent a probe attack. This coating is implemented with second–layer metal added through special processing of the microcontroller die. This additional layer is not a simple sheet of metal, but rather a complex layout that is interwoven with power and ground which are in turn connected to logic for the Encryption Key and the Security Lock. As a result, any attempt to remove the layer or probe through it will result in the erasure of the security lock and/or the loss of encryption key bits.

BOOTSTRAP LOADING

Initial loading of application software into the DS5002 is performed by firmware within the on–chip Bootstrap Loader communicating with a PC via the on–chip serial port in a manner which is almost identical to that for the DS5001. The user should consult the DS5001 data sheet as a basis of operational characteristics of this firmware. Certain differences in loading procedure exist in order to support the security feature. These differences are documented below. Table 1 summarizes the commands accepted by the bootstrap loader.

When the Bootstrap Loader is invoked, the 128–byte scratchpad RAM area is automatically overwritten with zeroes, and then used for variable storage for the bootstrap firmware. Also, a set of 8 bytes are generated using the random number generator circuitry and are saved as a potential word for the 64–bit encryption key.

Any read or write operation to the DS5002's external program/data SRAM can only take place if the Security Lock bit is in a cleared state. Therefore, the first step which is taken in the loading of a program should be the clearing of the Security Lock bit through the "U" command.

DS5002 SERIAL BOOTSTRAP LOADER COMMANDS Table 1

COMMAND	FUNCTION
C	Return CRC–16 of the program/data NVRAM
D	Dump Intel Hex file
F	Fill program/data NVRAM
G	Get Data from P1, P2, and P3
I	N/A on the DS5002FP
L	Load Intel Hex file
M	Toggle modem available bit
N	Set Freshness Seal - All program and data will be lost
P	Put data into P0, P1, P2, and P3
R	Read status of NVSFRs (MCON, RPCTL, MSL, CALIB)
T	Trace (echo) incoming Intel Hex code
U	Clear security lock
V	Verify program/data NVRAM with incoming Intel Hex data
W	Write Special Function Registers - (MCON, RPCTL, MSL, CALIB)
Z	Set security lock

Execution of certain bootstrap loader commands will result in the loading of the newly generated 64–bit random number into the encryption key word. These commands are as follows:

a. Fill F
b. Load L
c. Dump D
d. Verify V
e. CRC C

Execution of the Fill and Load commands will result in the data loaded into the NVRAM in an encrypted form determined by the value of the newly–generated key word. The subsequent execution of the Dump command within the same bootstrap session will cause the contents of the encrypted RAM to be read out and transmitted back to the host PC in decrypted form. Similarly, execution of the Verify command within the same bootstrap session will cause the incoming absolute hex data to be compared against the true contents of the encrypted RAM, and the CRC command will return the CRC value calculated from the true contents of the encrypted RAM. As long as any of the above commands are executed within the same bootstrap session, the loaded key value will remain the same and contents of the encrypted program/data NVRAM may be read or written normally and freely until the Security Lock bit is set.

When the Security Lock bit is set using the Z command, no further access to the true RAM contents is possible using any bootstrap command or by any other means.

INSTRUCTION SET

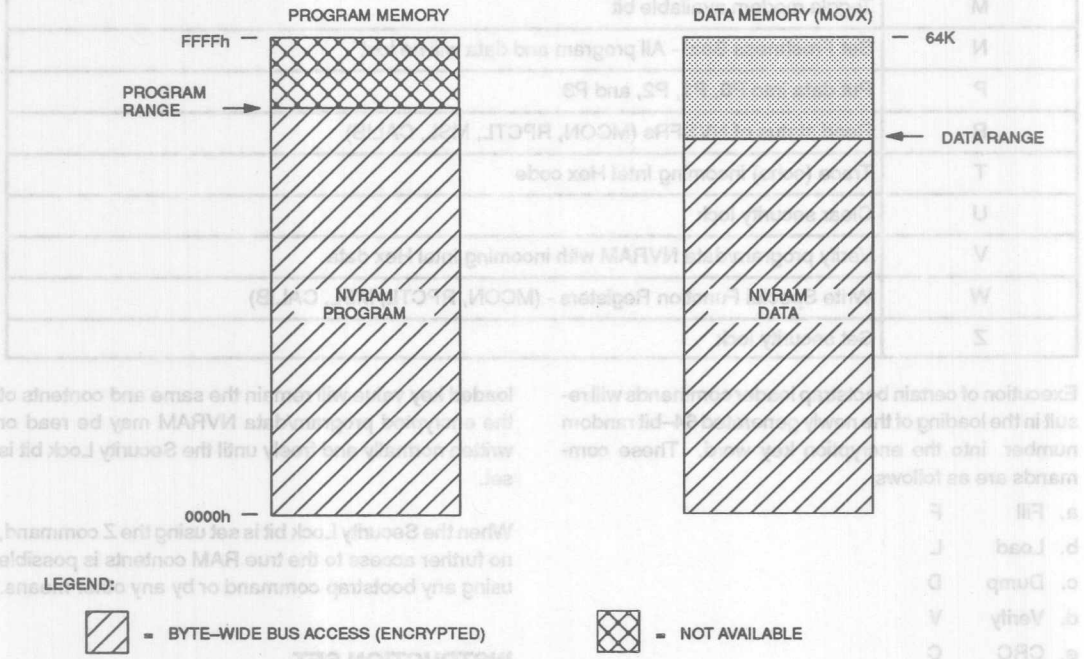
The DS5002FP executes an instruction set that is object code compatible with the industry standard 8051 microcontroller. As a result, software development packages such as assemblers and compilers that have been written for the 8051 are compatible with the DS5002FP. A complete description of the instruction set and operation are provided in the User's Guide section of the Soft Microcontroller Data Book.

Also note that the DS5002FP is embodied in the DS2252(T) module. The DS2252(T) combines the DS5002FP with between 32K and 128K of SRAM, and a lithium cell. An optional Real–time Clock is also available in the DS2252T. This is packaged in a 40–pin SIMM module.

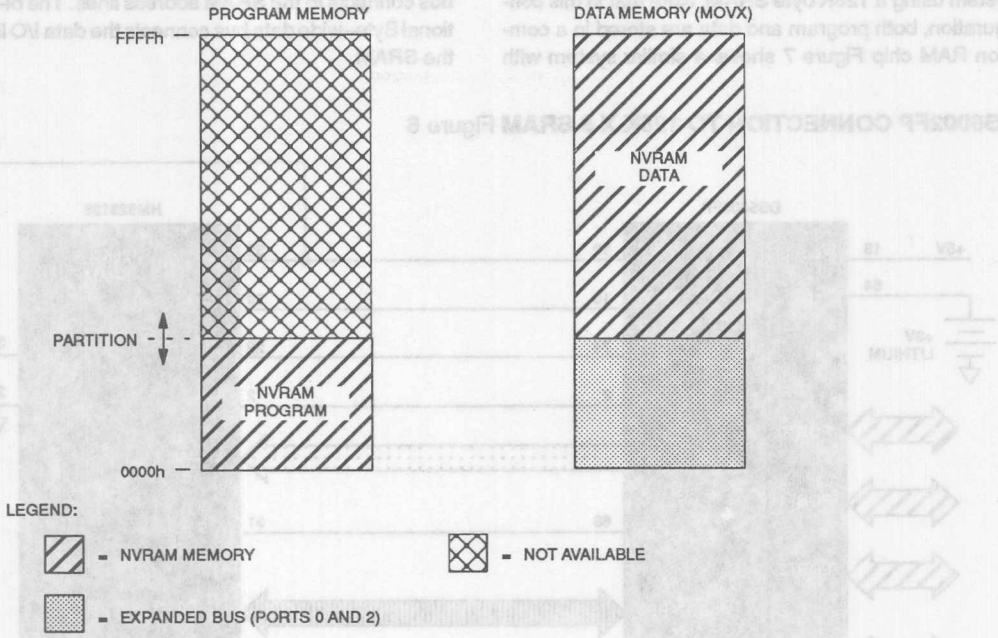
MEMORY ORGANIZATION

Figure 3 illustrates the memory map accessed by the DS5002FP. The entire 64K of program and 64K of data are potentially available to the Byte–wide bus. This preserves the I/O ports for application use. The user controls the portion of memory that is actually mapped to the Byte–wide bus by selecting the Program Range and Data Range. Any area not mapped into the NVRAM is reached via the Expanded bus on Ports 0 & 2. An alternate configuration allows dynamic Partitioning of a 64K space as shown in Figure 4. Selecting PES=1 provides another 64K of potential data storage or memory mapped peripheral space as shown in Figure 5. These selections are made using Special Function Registers. The memory map and its controls are covered in detail in the User's Guide section of the Soft Microcontroller Data Book.

MEMORY MAP OF THE DS5002FP WITH PM=1 Figure 3



MEMORY MAP OF THE DS5002FP WITH PM=0 Figure 4



MEMORY MAP OF THE DS5002FP WITH PES=1 Figure 5

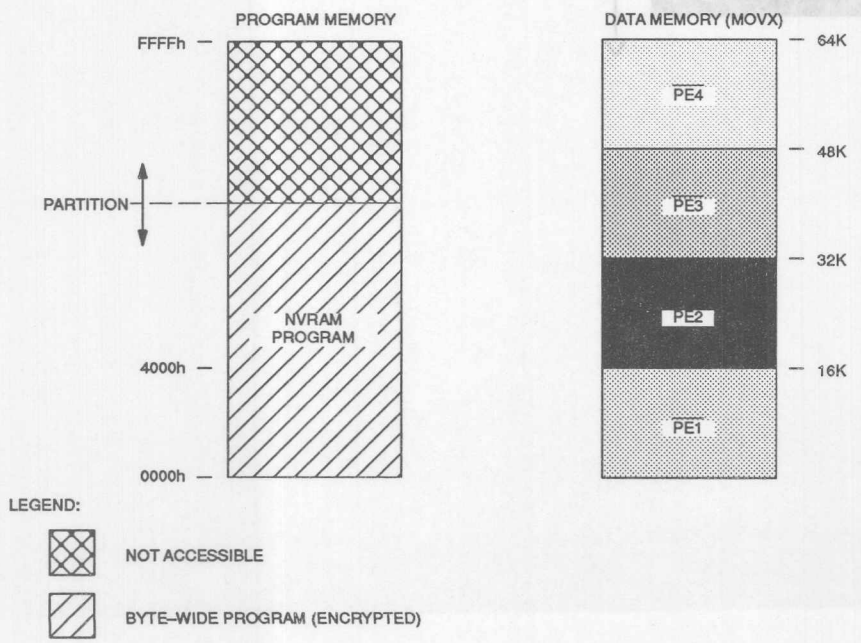
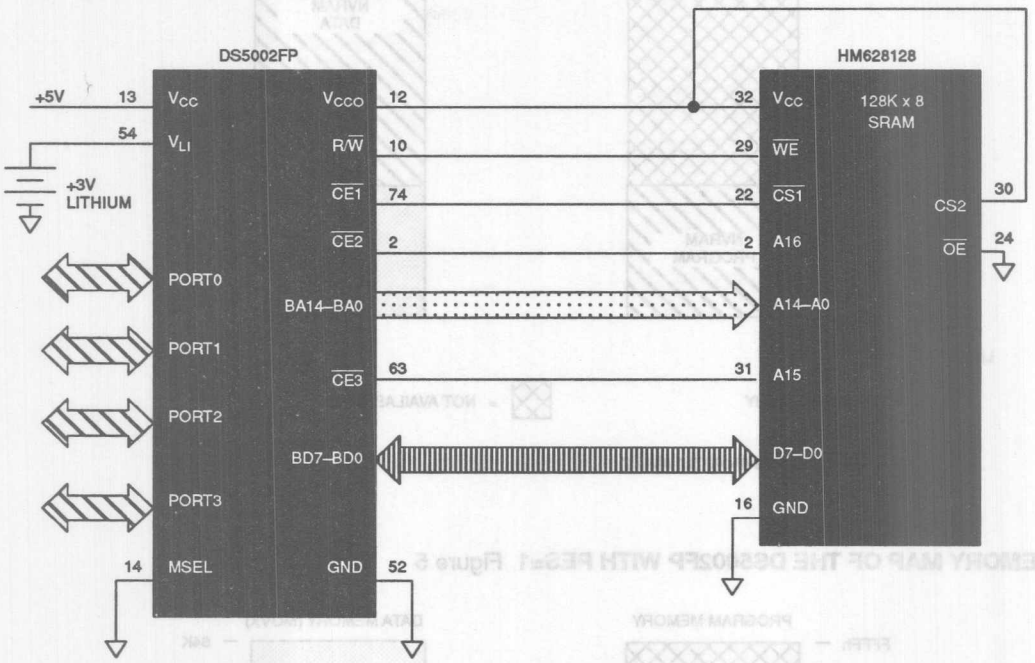
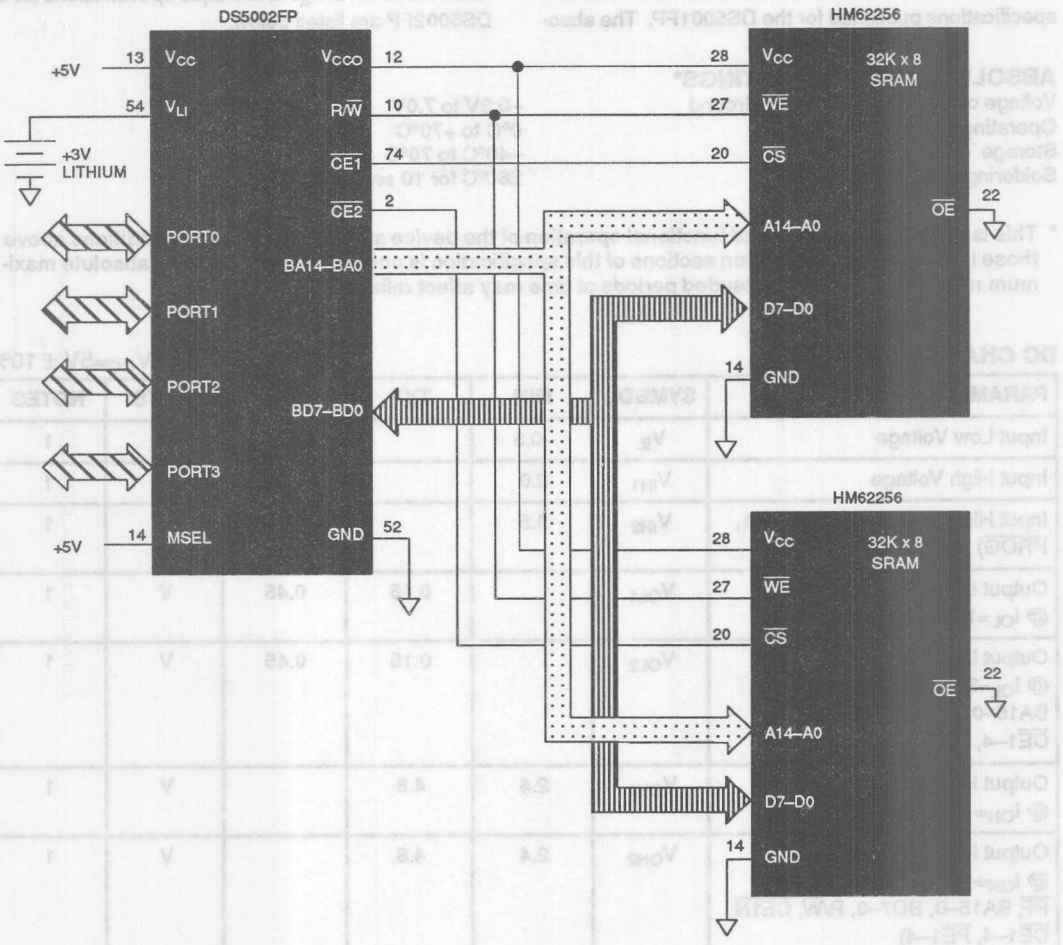


Figure 6 illustrates a typical memory connection for a system using a 128K byte SRAM. Note that in this configuration, both program and data are stored in a common RAM chip Figure 7 shows a similar system with using two 32K byte SRAMs. The Byte–wide Address bus connects to the SRAM address lines. The bi–directional Byte–wide data bus connects the data I/O lines of the SRAM.

DS5002FP CONNECTION TO 128K X 8 SRAM Figure 6



DS5002FP CONNECTION TO 64K X 8 SRAM Figure 7



POWER MANAGEMENT

The DS5002FP monitors V_{CC} to provide Power–fail Reset, early warning Power–fail Interrupt, and switch over to lithium backup. It uses an internal band–gap reference in determining the switch points. These are called V_{PFW}, V_{CCMIN}, and V_{LI} respectively. When V_{CC} drops below V_{PFW}, the DS5002FP will perform an interrupt vector to location 2Bh if the power fail warning was enabled. Full processor operation continues regardless. When power falls further to V_{CCMIN}, the DS5002FP invokes a reset state. No further code execution will be performed unless power rises back above V_{CCMIN}. All decoded chip enables and the R/$\overline{W}$ signal go to an inactive (logic 1) state. V_{CC} is still the power source at this time. When V_{CC} drops further to below V_{LI}, internal circuitry will switch to the lithium cell for power. The majority of internal circuits will be disabled and the remaining nonvolatile states will be retained. Any devices connected to V_{CCO} will be powered by the lithium cell at this time. V_{CCO} will be at the lithium battery voltage less a diode drop. This drop will vary depending on the load. Low power SRAMs should be used for this reason. When using the DS5002FP, the user must select the appropriate battery to match the RAM data retention current and the desired backup lifetime. Note that the lithium cell is only loaded when $V_{CC} < V_{LI}$. The User's Guide has more information on this topic. The trip points V_{CCMIN} and V_{PFW} are listed in the electrical specifications.

ELECTRICAL SPECIFICATIONS

The DS5002FP adheres to all AC and DC electrical specifications published for the DS5001FP. The absolute maximum ratings and unique specifications for the DS5002FP are listed below.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	–0.3V to 7.0V
Operating Temperature	0°C to +70°C
Storage Temperature	–40°C to 70°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

DC CHARACTERISTICS

(t_A=0°C to 70°C; V_{CC}=5V ± 10%)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Low Voltage	V_{IL}	–0.3		0.8	V	1
Input High Voltage	V_{IH1}	2.0		V_{CC}+0.3	V	1
Input High Voltage (RST, XTAL1, $\overline{PROG}$)	V_{IH2}	3.5		V_{CC}+0.3	V	1
Output Low Voltage @ I_{OL}=1.6 mA (Ports 1, 2, 3)	V_{OL1}		0.15	0.45	V	1
Output Low Voltage @ I_{OL}=3.2mA (Port 0, ALE, $\overline{PF}$, BA15–0, BD7–0, R/$\overline{W}$, $\overline{CE1N}$, $\overline{CE1}$–4, $\overline{PE1}$–4)	V_{OL2}		0.15	0.45	V	1
Output High Voltage @ I_{OH}=–80 mA (Ports 1, 2, 3)	V_{OH1}	2.4	4.8		V	1
Output High Voltage @ I_{OH}=–400 µA (Ports 0, ALE, $\overline{PF}$, BA15–0, BD7–0, R/$\overline{W}$, $\overline{CE1N}$, $\overline{CE1}$–4, $\overline{PE1}$–4)	V_{OH2}	2.4	4.8		V	1
Input Low Current V_{IN}=0.45V (Ports 1, 2, 3)	I_{IL}			–50	µA	
Transition Current; 1 to 0 V_{IN}=2.0V (Ports 1, 2, 3)	I_{TL}			–500	µA	
SDI Input Low Voltage	V_{ILS}			0.4	V	1
SDI Input High Voltage	V_{IHS}	2.0		V_{CCO}	V	1
SDI Pull–Down Resistor	R_{SDI}	25		60	KΩ	
Battery–Backup Quiescent Current	I_{BAT}		5	75	nA	7

DC CHARACTERISTICS (cont'd)

(t_A=0°C to 70°C; V_{CC}=5V ± 10%)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Leakage Current 0.45<V_{IN},V_{CC} (Port 0, MSEL)	I_{IL}			±10	µA	
RST Pulldown Resistor	R_{RE}	40		150	KΩ	
$\overline{VRST}$ Pullup Resistor	R_{VR}		4.7		KΩ	
$\overline{PROG}$ Pullup Resistor	R_{PR}		40		KΩ	
Power Fail Warning Voltage	V_{PRW}	4.25	4.37	4.50	V	1
Minimum Operating Voltage	V_{CCMIN}	4.00	4.12	4.25	V	1
Lithium Supply Voltage	V_{IL}	2.5		4.0	V	1
Operating Current	I_{CC}			36	mA	2
Idle Mode Current	I_{IDLE}			7.0	mA	3
Stop Mode Current	I_{STOP}			80	µA	4
Pin Capacitance	C_{IN}			10	pF	5
Output Supply Voltage (V_{CCO})	V_{CCO1}			V_{CC}–0.3	V	1, 2
Output Supply Battery–backed Mode (V_{CCO}, $\overline{CE}$1–4, $\overline{PE}$1–2)	V_{CCO2}			V_{LI}–0.55	V	1,8
Output Supply Current @ V_{CCO}=V_{CC} – 0.3V	I_{CCO1}			75	mA	6
Lithium–backed Quiescent Current	I_{LI}		5	75	nA	7
Reset Trip Point in Stop Mode w/BAT=3.0V w/BAT=3.3V		 4.0 4.4		 4.25 4.65		1

AC CHARACTERISTICS

(t_A=0°C to70°C; V_{CC}=0V to 5V)

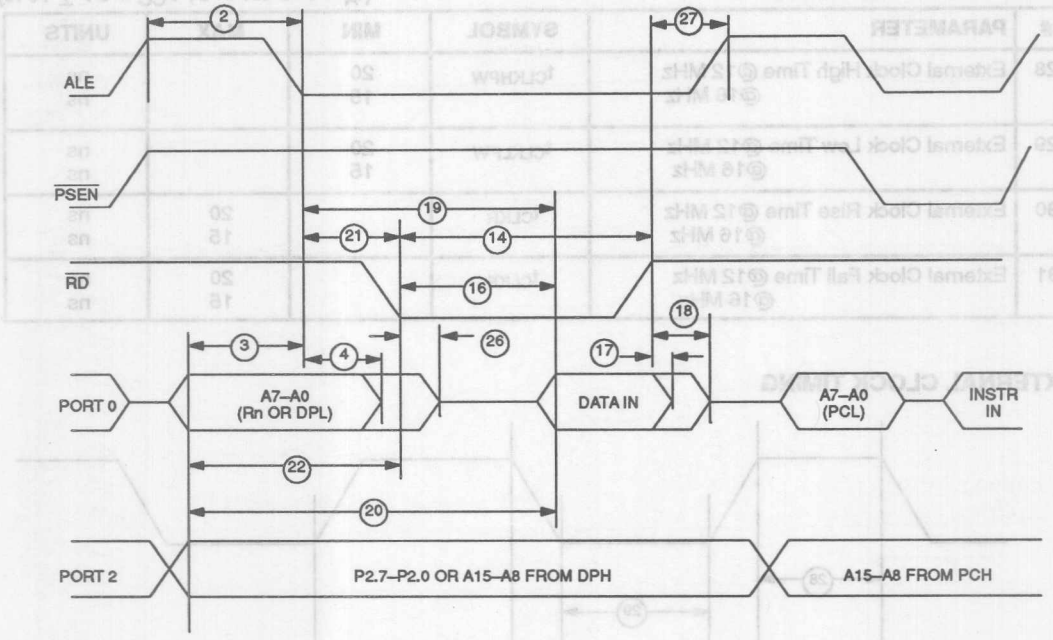
PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
SDI Pulse Reject	t_{SPR}			2	us	10
SDI Pulse Accept	t_{SPA}	10			us	10

AC CHARACTERISTICS
EXPANDED BUS MODE TIMING SPECIFICATIONS

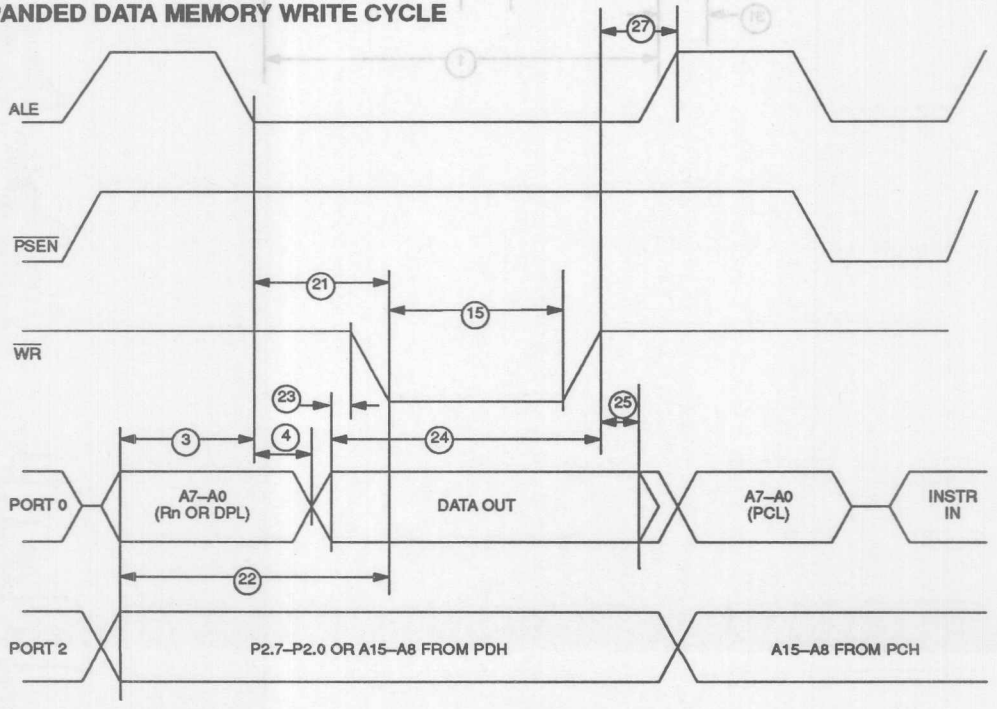
(t_A=0°C to70°C; V_{CC}=5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
1	Oscillator Frequency	$1/t_{CLK}$	1.0	12 (-12) 16 (-16)	MHz
2	ALE Pulse Width	t_{ALPW}	$2t_{CLK}$-40		ns
3	Address Valid to ALE Low	t_{AVALL}	t_{CLK}-40		ns
4	Address Hold After ALE Low	t_{AVAAV}	t_{CLK}-35		ns
14	$\overline{RD}$ Pulse Width	t_{RDPW}	$6t_{CLK}$-100		ns
15	$\overline{WR}$ Pulse Width	t_{WRPW}	$6t_{CLK}$-100		ns
16	$\overline{RD}$ Low to Valid Data In @12 MHz @16 MHz	t_{RDLDV}		$5t_{CLK}$-165 $5t_{CLK}$-105	ns ns
17	Data Hold after $\overline{RD}$ High	t_{RDHDV}	0		ns
18	Data Float after $\overline{RD}$ High	t_{RDHDZ}		$2t_{CLK}$-70	ns
19	ALE Low to Valid Data In @12 MHz @16 MHz	t_{ALLVD}		$8t_{CLK}$-150 $8t_{CLK}$-90	ns ns
20	Valid Addr. to Valid Data In @12 MHz @16 MHz	t_{AVDV}		$9t_{CLK}$-165 $9t_{CLK}$-105	ns ns
21	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	t_{ALLRDL}	$3t_{CLK}$-50	$3t_{CLK}$+50	ns
22	Address Valid to $\overline{RD}$ or $\overline{WR}$ Low	t_{AVRDL}	$4t_{CLK}$-130		ns
23	Data Valid to $\overline{WR}$ Going Low	t_{DVWRL}	t_{CLK}-60		ns
24	Data Valid to $\overline{WR}$ High @12 MHz @16 MHz	t_{DVWRH}	$7t_{CLK}$-150 $7t_{CLK}$-90		ns ns
25	Data Valid after $\overline{WR}$ High	t_{WRHDV}	t_{CLK}-50		ns
26	$\overline{RD}$ Low to Address Float	t_{RDLAZ}		0	ns
27	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{RDHALH}	t_{CLK}-40	t_{CLK}+50	ns

EXPANDED DATA MEMORY READ CYCLE



EXPANDED DATA MEMORY WRITE CYCLE

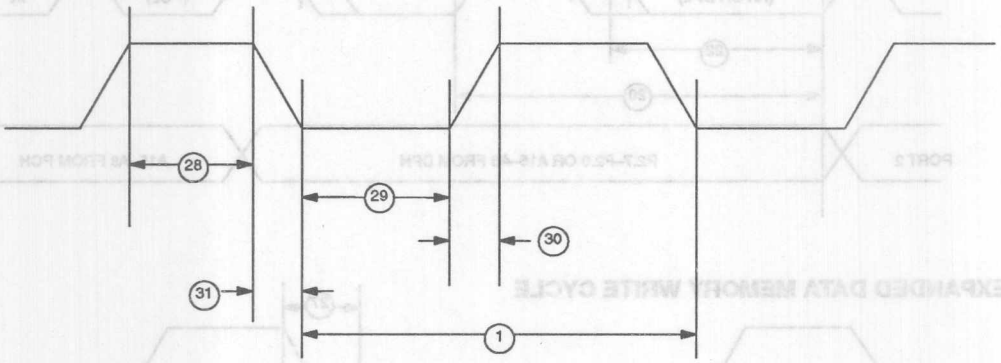


AC CHARACTERISTICS (cont'd)
EXTERNAL CLOCK DRIVE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
28	External Clock High Time @12 MHz @16 MHz	t_{CLKHPW}	20 15		ns ns
29	External Clock Low Time @12 MHz @16 MHz	t_{CLKLPW}	20 15		ns ns
30	External Clock Rise Time @12 MHz @16 MHz	t_{CLKR}		20 15	ns ns
31	External Clock Fall Time @12 MHz @16 MHz	t_{CLKF}		20 15	ns ns

EXTERNAL CLOCK TIMING



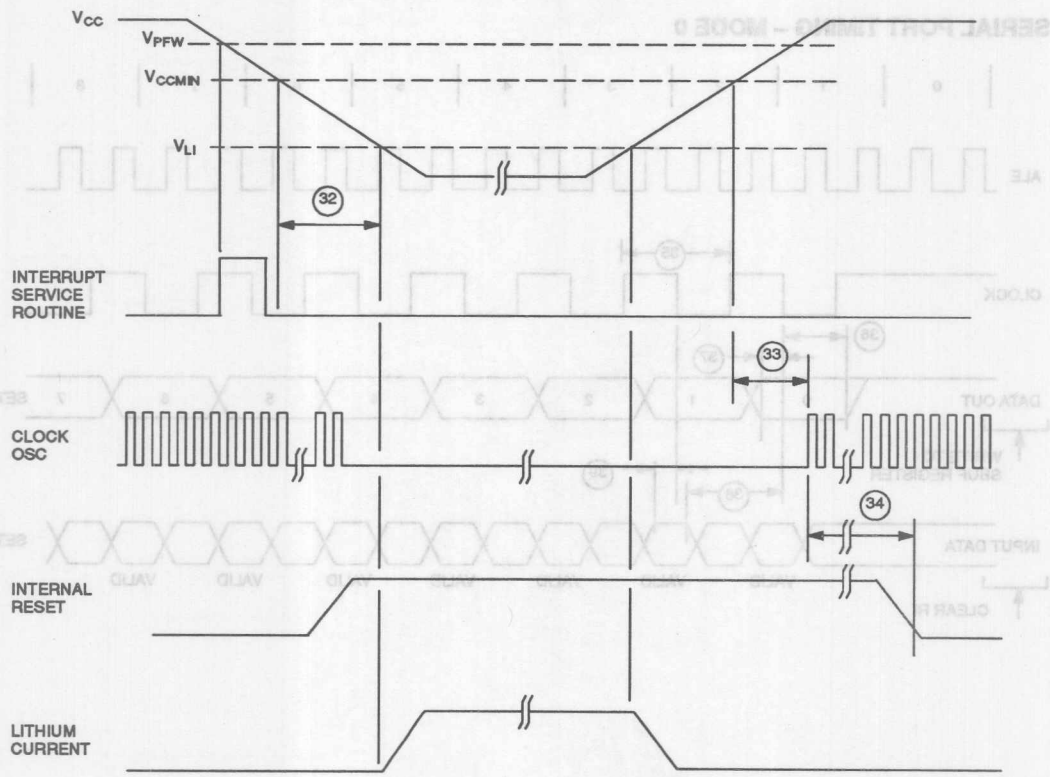
AC CHARACTERISTICS (cont'd)

POWER CYCLING TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
32	Slew Rate from V_{CCmin} to V_{LI}	t_F	130		µs
33	Crystal Start up Time	t_{CSU}		(note 9)	
34	Power On Reset Delay	t_{POR}		21504	t_{CLK}

POWER CYCLE TIMING

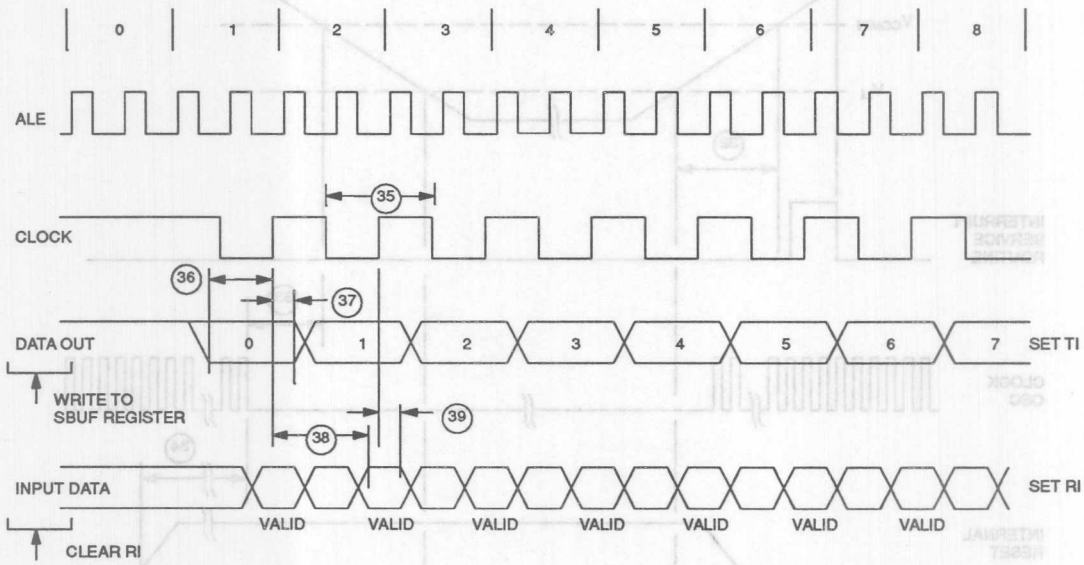


AC CHARACTERISTICS (cont'd)
SERIAL PORT TIMING – MODE 0

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
35	Serial Port Clock Cycle Time	t_{SPCLK}	$12t_{CLK}$		µs
36	Output Data Setup to Rising Clock Edge	t_{DOCH}	$10t_{CLK}$-133		ns
37	Output Data Hold after Rising Clock Edge	t_{CHDO}	$2t_{CLK}$-117		ns
38	Clock Rising Edge to Input Data Valid	t_{CHDV}		$10t_{CLK}$-133	ns
39	Input Data Hold after Rising Clock Edge	t_{CHDIV}	0		ns

SERIAL PORT TIMING – MODE 0

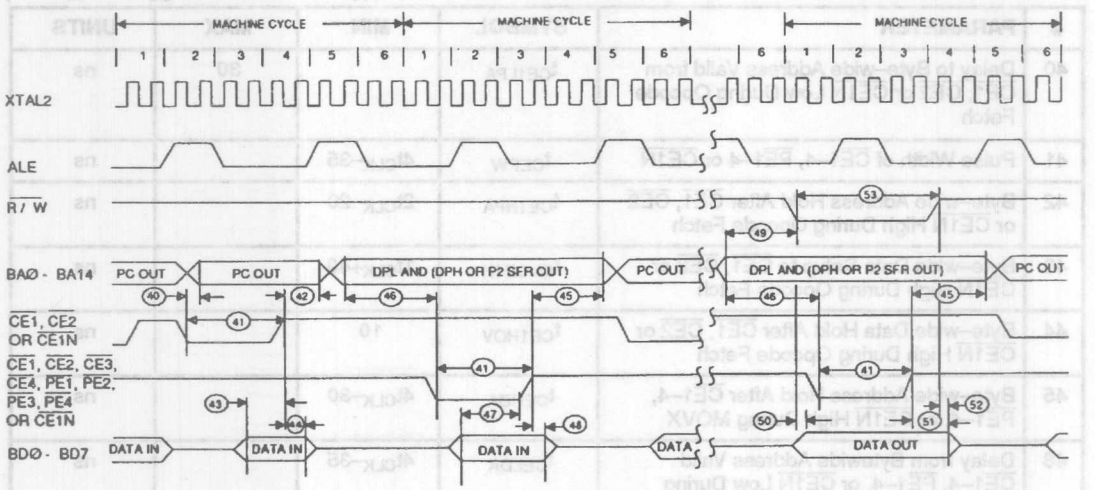


AC CHARACTERISTICS
BYTEWIDE ADDRESS/DATA BUS TIMING

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
40	Delay to Byte–wide Address Valid from $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ Low During Opcode Fetch	t_{CE1LPA}		30	ns
41	Pulse Width of $\overline{CE1}$–4, $\overline{PE1}$–4 or $\overline{CE1N}$	t_{CEPW}	$4t_{CLK}$–35		ns
42	Byte–wide Address Hold After $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HPA}	$2t_{CLK}$–20		ns
43	Byte–wide Data Setup to $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{OVCE1H}	$1t_{CLK}$+40		ns
44	Byte–wide Data Hold After $\overline{CE1}$, $\overline{CE2}$ or $\overline{CE1N}$ High During Opcode Fetch	t_{CE1HOV}	10		ns
45	Byte–wide Address Hold After $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX	t_{CEHDA}	$4t_{CLK}$–30		ns
46	Delay from Bytewide Address Valid $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ Low During MOVX	t_{CELDA}	$4t_{CLK}$–35		ns
47	Bytewide Data Setup to $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{DACEH}	$1t_{CLK}$+40		ns
48	Bytewide Data Hold After $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High During MOVX (read)	t_{CEHDV}	10		ns
49	Bytewide Address Valid to R/$\overline{W}$ Active During MOVX (write)	t_{AVRWL}	$3t_{CLK}$–35		ns
50	Delay from R/$\overline{W}$ Low to Valid Data Out During MOVX (write)	t_{RWLDV}	20		ns
51	Valid Data Out Hold Time from $\overline{CE1}$–4, $\overline{PE1}$–4, or $\overline{CE1N}$ High	t_{CEHDV}	$1t_{CLK}$–15		ns
52	Valid Data Out Hold Time from R/$\overline{W}$ High	t_{RWHDV}	0		ns
53	Write Pulse Width (R/$\overline{W}$ Low Time)	t_{RWLPW}	$6t_{CLK}$–20		ns

BYTEWIDE BUS TIMING



RPC AC CHARACTERISTICS – DBB READ

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
54	$\overline{CS}$, A_0 Setup to $\overline{RD}$	t_{AR}	0		ns
55	$\overline{CS}$, A_0 Hold After $\overline{RD}$	t_{RA}	0		ns
56	$\overline{RD}$ Pulse Width	t_{RR}	160		ns
57	$\overline{CS}$, A_0 to Data Out Delay	t_{AD}		130	ns
58	$\overline{RD}$ to Data Out Delay	t_{RD}	0	130	ns
59	$\overline{RD}$ to Data Float Delay	t_{RDZ}		85	ns

RPC AC CHARACTERISTICS – DBB WRITE

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
60	$\overline{CS}$, A_0 Setup to $\overline{WR}$	t_{AW}	0		ns
61A	$\overline{CS}$, Hold After $\overline{WR}$	t_{WA}	0		ns
61B	A_0, Hold After $\overline{WR}$	t_{WA}	20		ns
62	$\overline{WR}$ Pulse Width	t_{WW}	160		ns
63	Data Setup to $\overline{WR}$	t_{DW}	130		ns
64	Data Hold After $\overline{WR}$	t_{WD}	20		ns

AC CHARACTERISTICS – DMA

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

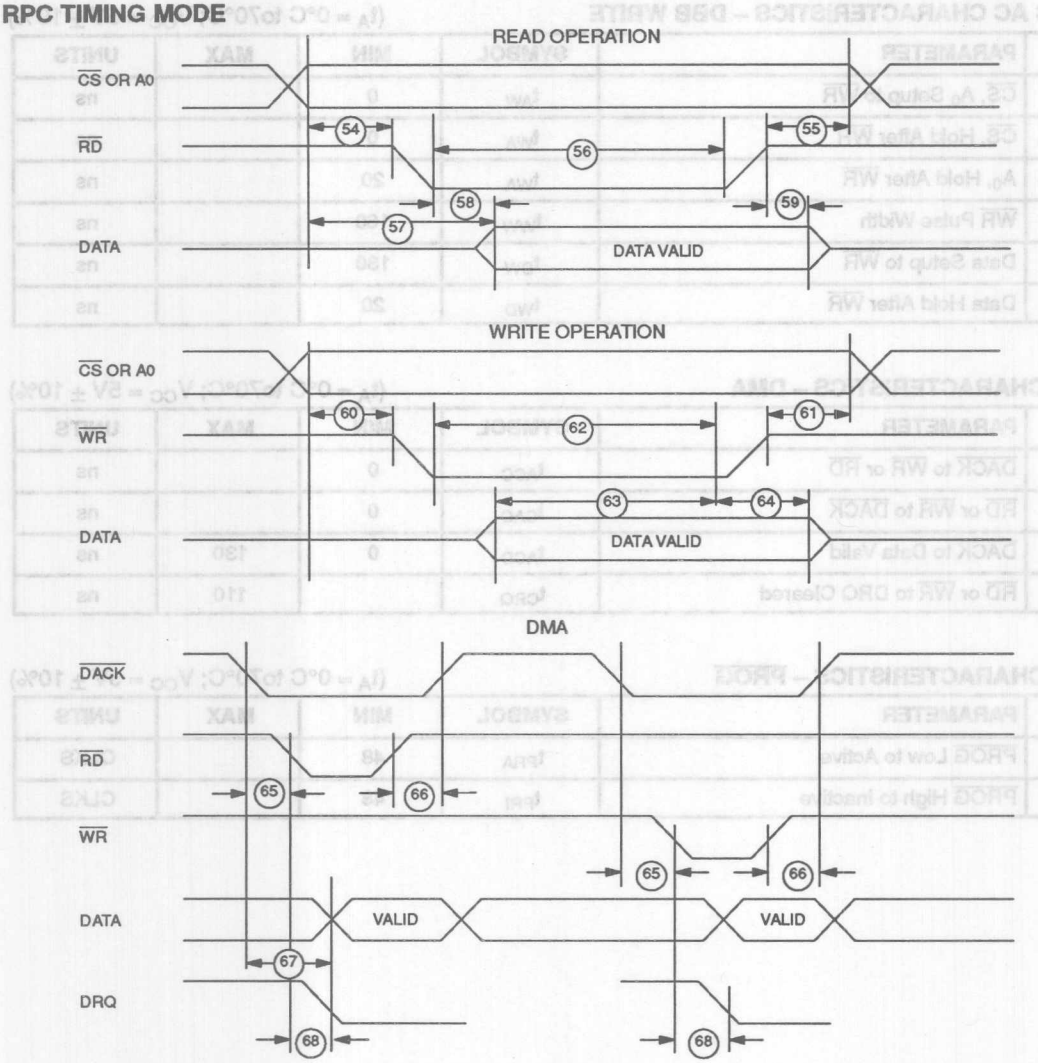
#	PARAMETER	SYMBOL	MIN	MAX	UNITS
65	$\overline{DACK}$ to $\overline{WR}$ or $\overline{RD}$	t_{ACC}	0		ns
66	$\overline{RD}$ or $\overline{WR}$ to $\overline{DACK}$	t_{CAC}	0		ns
67	$\overline{DACK}$ to Data Valid	t_{ACD}	0	130	ns
68	$\overline{RD}$ or $\overline{WR}$ to DRQ Cleared	t_{CRQ}		110	ns

AC CHARACTERISTICS – $\overline{PROG}$

(t_A = 0°C to70°C; V_{CC} = 5V ± 10%)

#	PARAMETER	SYMBOL	MIN	MAX	UNITS
69	$\overline{PROG}$ Low to Active	t_{PRA}	48		CLKS
70	$\overline{PROG}$ High to Inactive	t_{PRI}	48		CLKS

RPC TIMING MODE

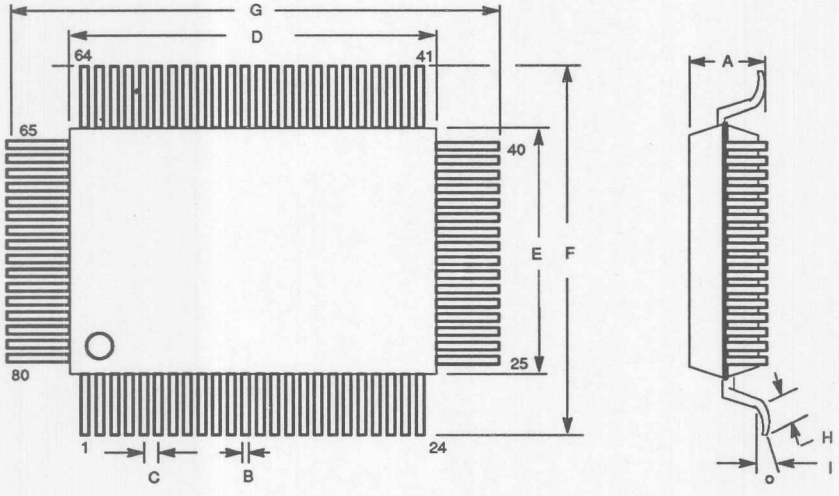


NOTES:

1. All voltages are referenced to ground.
2. Maximum operating I_{CC} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF}=10 ns, V_{IL} = 0.5V; XTAL2 disconnected; RST = PORT0 = V_{CC}, MSEL = V_{SS}.
3. Idle mode I_{IDLE} is measured with all output pins disconnected; XTAL1 driven with t_{CLKR}, t_{CLKF} = 10 ns, V_{IL} = 0.5V; XTAL2 disconnected; PORT0 = V_{CC}, RST = MSEL = V_{SS}.
4. Stop mode I_{STOP} is measured with all output pins disconnected; PORT0 = V_{CC}; XTAL2 not connected; RST = MSEL = XTAL1 = V_{SS}.
5. Pin Capacitance is measured with a test frequency – 1 MHz, t_A = 25°C.

6. I_{CCO1} is the maximum average operating current that can be drawn from V_{CCO} in normal operation.
7. I_{LI} is the current drawn from V_{LI} input when V_{CC} = 0V and V_{CCO} is disconnected. Battery–backed mode: 2.5V ≤ V_{BAT} ≤ 4.0; V_{CC} ≤ V_{BAT}; V_{SDI} should be ≤ V_{ILS} for I_{BAT} max.
8. V_{CCO2} is measured with V_{CC} < V_{LI}, and a maximum load of 10 µA on V_{CCO}.
9. Crystal start–up time is the time required to get the mass of the crystal into vibrational motion from the time that power is first applied to the circuit until the first clock pulse is produced by the on–chip oscillator. The user should check with the crystal vendor for a worst case specification on this time.
10. SDI is deglitched to prevent accidental destruction. The pulse must be longer than t_{SPR} to pass the deglitcher, but SDI is not guaranteed unless it is longer than t_{SPA}.

DS5002FP CMOS MICROCONTROLLER



DIM	MILLIMETERS		
	MIN	NOM	MAX
A	–	2.91	3.15
B	0.25	0.35	0.45
C	–	0.80	–
D	19.85	20.00	20.15
E	13.85	14.00	14.15
F	17.40	17.86	18.20
G	23.40	23.86	24.20
H	0.40	–	1.3
I	0	–	10°

DIM	MILLIMETERS		
	MIN	NOM	MAX
A	–	2.91	3.15
B	0.25	0.35	0.45
C	–	0.80	–
D	19.85	20.00	20.15
E	13.85	14.00	14.15
F	17.40	17.86	18.20
G	23.40	23.86	24.20
H	0.40	–	1.3
I	0	–	10°

DEVELOPMENT TOOLS

DEVELOPMENT SUPPORT

Developing a system based on the Soft Microcontroller is easy. A range of 8051 tools are available from third party suppliers, and these are compatible with all Soft Micro products. Selected tools are available that support the Soft Micro, though many systems are developed using only standard 8051 tools.

ASSEMBLERS

The Soft Micro runs 8051 instruction set and is object code compatible. This means that standard 8051 assemblers will generate the correct code. There is no need for a special assembler. The Soft Micro does have numerous extra features, but no extra instructions. The unique features are all accessed via Special Function Registers. These should be defined in the user's software via equate statements. In this way, the user identifies the new registers, allowing the full range of instruction operations. Once defined by the user, the new SFR will receive the same treatment as any of the standard ones. The assembler has no knowledge of the actual hardware associated with any SFR.

An Intel Hex file format is required by the Bootstrap Loader built into each Soft Micro. This is a standard out put file format for all 8051 compatible assemblers. Note, a utility may be required to create the Intel hex file. That is, some assemblers generate a binary file as the standard output. This file may then pass through a utility that converts object to hex. In all cases, the utility will be included with the assembler.

For the convenience of the user, the following is a list of the SFRs that are unique to the Soft Micro family and their equate locations. Simply include the relevant values in the program source file.

Name	Symbol	Equate Location
RPC Status	STATUS	0DAh
RPC Control	RPCTL	0D8h
Random Number	RNR	0CFh
Timed Access	TA	0C7h
Memory Control	MCON	0C6h
CRC High Value	CRC_HIGH	0C3h
CRC Low Value	CRC_LOW	0C2h
CRC Control	CRC	0C1h

HIGH LEVEL LANGUAGE COMPILERS

There are two languages that are commonly used in microcontroller system design. They are BASIC and C.

Although BASIC interpreters have been created for the 8051 family, a compiler is normally preferable. A BASIC compiler specifically for the Soft Micro is available from Systronix. They can be reached at 801–487–7412. The Systronix package supports Soft Micro memory mapping, Watchdog Timer, Real–time Clock and other features in addition to BASIC language support for the 8051 family.

There are no C compilers dedicated to the Soft Micro family at the time of this printing. However, like assembly language, standard 8051 C compilers can be used. The compiler must be informed of the existence and location of new SFRs that it will be accessing. There is one important note. When using C, it is commonly necessary to identify the starting address for various read/write segments such as XDATA and STACK. This may be done using a start–up file. When using a Soft Micro with a Partitioned memory map, the default value of 0000h for such segments is not advisable. The NVRAM area will begin at the logical Partition address. Therefore, the segments that require RAM beyond the 128 scratchpad locations should be located in the memory map at an address that is above the Partition. For example, if the Partition is located at address 4000, this is also a suitable beginning address for the STACK.

PROGRAMMERS

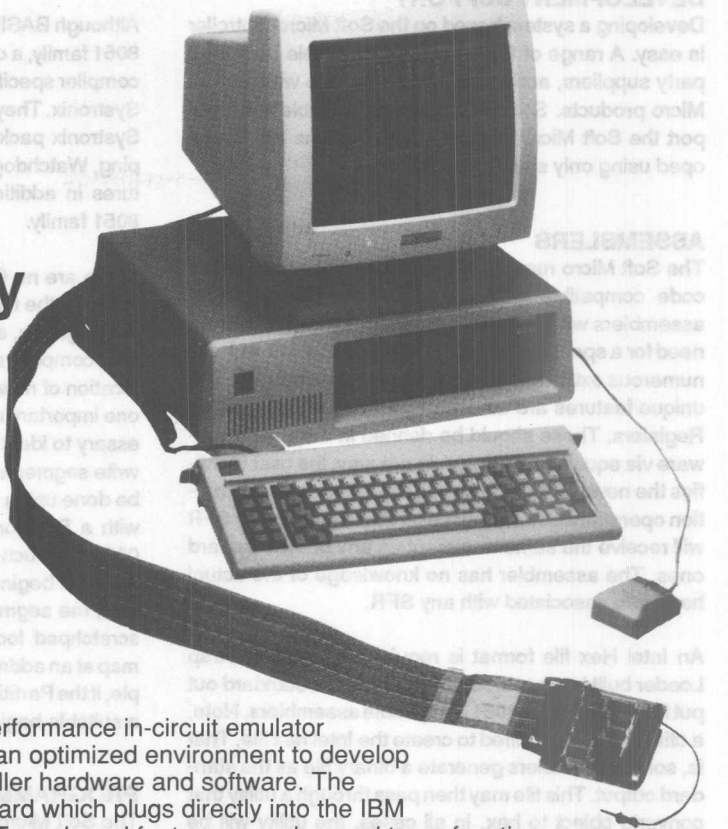
The Soft Micro provides its own built–in programming support with its Bootstrap Loader. This allows the micro to be programmed while installed in system or in a simple user fixture. Serial programming is preferred as a general technique as the easiest and fastest method. The DS5000TK evaluation kit discussed below can also be used a programmer.

EMULATORS

A Soft Micro emulator is available from Nohau Corp. They can be reached at 408–866–1820. The Nohau emulator provides development support for the 80–pin QFP version of the DS5000FP or DS5001FP. Using the QFP, a breadboard can be constructed of the user's system, even if the end version will use a module. Simply follow the guidelines for memory connection in the User's Guide section of the Soft Micro data book.

nohau
CORPORATION

8051 Family In-Circuit Emulator



The EMUL51-PC is a high performance in-circuit emulator specifically designed to give an optimized environment to develop your 8051 family microcontroller hardware and software. The EMUL51-PC consists of a board which plugs directly into the IBM PC/XT/AT bus. The optional Trace board features an advanced trace function with sophisticated trigger capabilities.
The POD, which plugs into the target system, is connected with a 5 ft. ribbon cable to the emulator board to provide a flexible operating range.
Optionally an RS-232 box can be used. It communicates with the PC at up to 115K baud.

User friendly

The friendly user interface is one of the key features of the EMUL51-PC. Pull-down menus, mouse support and on-line help make the infrequent user feel instantly at home. The more experienced user will appreciate the fast command line interpreter and the comprehensive command set with powerful macros.
The data windows on the screen contain up-to date information on symbols and memory areas. All commands are supported with context sensitive help information. If more help is needed the whole manual is on-line, available with a key stroke.
As an alternative, a Borland key press compatible user interface is available from ChipTools.

High level debugging

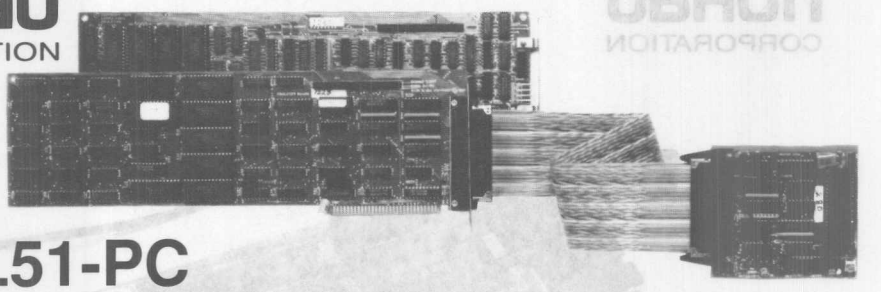
Using a high-level language for code generation is a way to cut development time. EMUL51-PC gives full support for debugging directly in C or PL/M source code. This eliminates tedious symbolic debugging in assembler.
With the EMUL51-PC you can mark breakpoints directly in the source code window, and it's easy to follow the program execution on the screen listing. All C variables, both local and global, can easily be displayed or altered in one of the dedicated windows. All variables are displayed in the form they were declared in your software. This way you can work directly with floats, arrays, pointers, structures, etc.

Real time Trace

The EMUL51-PC offers trace features not found in other emulators. For instance the trace buffer can record up to 256K bus cycles with 64 bits of data. The trace can be operated "on-the-fly" which means that it can be viewed, programmed and retriggered without disturbing program execution. With the trace set-up menu you can define exactly what events are to be stored in the trace buffer. The real-time trace can be stopped (triggered) at a selected event or after a combination of multiple events. By aligning the trigger point anywhere in the trace buffer you have a full choice of pre- and post triggering.

Nohau Corp. 51 E. Campbell Avenue Campbell, CA 95008 (408) 866-1820 FAX: (408) 378-7869

nohau
CORPORATION



EMUL51-PC

Saves time on all 8051 family projects.

EMUL51-PC is an in-circuit emulator dedicated to the 8051 family. The EMUL51-PC truly emulates the microcontrollers from all manufacturers of 8051 derivatives. This means that your EMUL51-PC will work just like your microcontroller when it is plugged into your target.

System Specification

Host
IBM PC/XT/AT, PS/2 or compatible.
Minimum 640K of RAM.
Monochrome, CGA, EGA or VGA in 25, 43 or 50 line mode.

External Box
The emulator boards can be installed in an external box with serial communication to PC. Up to 115K baud supported.

Languages supported
Third party assemblers, PL/M-51 and C-51 compilers.

High level debugging
Window for source level debugging. Single Step or Line Step with breakpoints marked directly in the code. Full support for local or global variables in C-51.

In-line Assembler and disassembler
Full instruction set and symbols supported.

Symbolic Support
Full symbolic debugging and type checking. Same symbols can be used in different modules.

File formats Supported
Intel HEX/OBJ/OMF/SYM, Avocet, Archimedes/IAR Franklin/Keil, BSO/Tasking, Intermetrics/Whitesmiths.

Real time Emulation
Full speed emulation up to 40 MHz.
No wait states and no intrusion on memory, stack, I/O or Interrupt pins.

Emulation Memory
64K XDATA memory and 64K CODE memory. Up to 256K with bankswitch option.

Memory Mapping
Mappable in 4K pages.

Macros
Test session automation and macro command definition. IF/ELSE, REPEAT/WHILE structures.

Debug Session Logging
Record emulation session and all setups to a file.

Breakpoints
64 K program breakpoints.
64 K data read and write breakpoints.
Break on external signal.
Break on direct access to internal bit or byte memory.
Break on a range of addresses.
With the trace board option you can break on any 48 bit combination of address, data, RD, WR, OP code fetch, interrupt level, ports of external signals.

Single Stepping
Single or multiple instruction stepping. Step over calls and interrupts.
Line stepping in high level languages.

Execution timer
Resolution down to half a cycle.

Real Time Trace (optional)
256K deep by 64 bits wide with time stamp.

Trace operation
At each cycle of execution 48 bits of address, data, ports and external signals are compared with eight independent "48 bit registers" to produce eight "condition signals". These eight conditions are then combined with six bits from a "64 state state-machine". The combination of these 14 signals creates a new state (or retains the old state). It also produces a TRIG and a FILTER signal that are used to control what is saved in the trace buffer. If the eight condition signals are named A, B, C, D, E, F, G and H an example of how this could be used follows:
A THEN B THEN C THEN D THEN E THEN F THEN G THEN H where A - H could be code addresses.
The trace buffer can be viewed, reprogrammed and restarted without affecting emulation.

Trigger Breakpoint
The 48 bit trace trigger qualifiers can be used to define complex breakpoints also on program execution.

Trace Display
Display trace in disassembled symbolic or binary/hex form, or as high level source code.
Display can be saved to a file.
Trace can be started, stopped and displayed independent of program execution.

Program Performance Analyzer
Histogram and statistical information of program execution in real time.

Supported chips

DS5000
DS5000FP
DS5000T
DS5001FP
DS5002FP

Packages supported

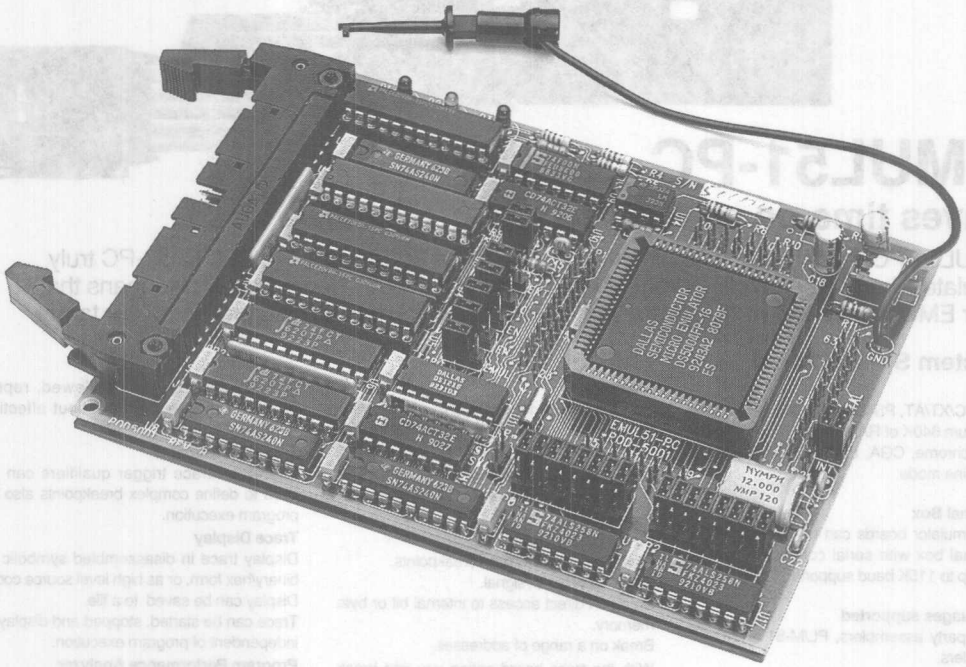
40-pin Dual Inline Package
80-pin Quad Flat Pack

Future products

Call Nohau Corporartion about plans for emulation of Dallas Semiconductor's DS80C320.

Nohau Corp. 51 E. Campbell Avenue Campbell, CA 95008 (408) 866-1820 FAX: (408) 378-7869

nohau
CORPORATION



POD-5001-16

The POD-5001-16 supports the **DS5000, DS5000FP, DS5000T, DS5001FP** and **DS5002FP**. Use it together with a Nohau emulator board, such as the EMUL51-PC/E128-16. The emulator board is a PC plug-in board and connects to the POD-5001-16 with a five-foot (1.5 m) cable. Jumpers on the POD allow selection of special Dallas microcontroller features, such as the clock of the **DS5000T** or the security features of the **DS5002FP**. Optional standard and advanced trace buffer boards and external RS-232 box systems are also available.

The POD includes an adapter, ADAP5001-DIP40, to plug into a 40-pin user target socket. An optional adapter, ET/EPP-080-QF08-LG, is available to solder to the user target board's 80-pin quad flat pack (QFP) pattern.

Jumpers let the user select the clock source on his own board or the on-board quartz crystal. The emulator and POD can operate standalone, without being plugged into a user target system.

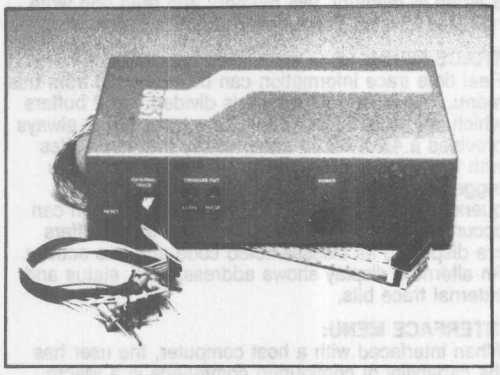
ET/ EPP-080-QF08-LG

Adapter for 80-pin QFP. The bottom of this adapter solders to user target board. The POD-5001-16 plugs into the top of the adapter.

Nohau Corp. 51 E. Campbell Avenue Campbell, CA 95008 (408) 866-1820 FAX: (408) 378-7869

HMI

HMI-200 SERIES
IN-CIRCUIT
EMULATOR



OPTIONAL MODULE AVAILABLE
TO SUPPORT THE DS5000

8051 Family In-Circuit Emulator (HMI-200-8051)

FEATURES

- Real-time emulation up to 16 MHz and single step.
- Four events for break and trigger operations can be individually configured as a bit pattern including don't care conditions of address, data I/O Ports, status, 16 external trace/trigger lines, and a pass counter of 1-65535 counts.
- The address and data fields of each event can be used to generate range break and trigger operations.
- Event operations can be programmed to occur from sequences. e.g. A then B then C then D.
- 128K bytes of emulation memory standard, with a battery back-up option available.
- Two 4K X 88 bit trace buffers.
- Trace qualification to allow selective tracing.
- Freeze trace allows the trace buffer to be viewed during emulation and new trigger conditions to be set.
- Macros allow a series of commands to be executed under one command name.
- A printer can be connected to the auxiliary serial port for hard copies of the trace buffer.
- User interface features the option of menu mode or command line mode for optimum speed and convenience.
- An interval timer measures the elapsed time from one event sequence to another.
- Two RS232 ports, supporting baud rates up to 38.4K baud.
- Binary file transfer between host computer and emulator for optimum speed in file transfer.
- User defined system configurations are stored in non-volatile memory.
- Symbolic and source level debuggers are available to run on IBM PC family or UNIX systems.
- Optional 256K bytes of symbol storage for stand-alone operation i.e. perform symbolic debugging while connected to a terminal
- Pass-thru command allows a terminal to communicate thru the emulator to a host computer (e.g. a VAX). After downloading executable code and symbol tables, the emulator can operate strictly from the terminal, completely independent of the host computer.

APPLICATIONS

- Supports the Intel 8051/8031, 8052/8032, OKI 80C154, and AMD 8053. Also supports the CMOS versions of these processors.
- Automatic test equipment.
- Hardware and software development, debugging and testing.
- Field troubleshooting aid.

DESCRIPTION

The HMI-200-8051 emulator is a high performance development system which combines the control of an in-circuit emulator with the power of a logic analyzer to provide a complete debugging environment for hardware and software on microprocessor based systems.

The operation of the system is performed from six convenient menus.

CONFIGURATION MENU:
From this menu, the user can configure the system parameters such as selecting target system crystal/clock or emulator cyrstal/clock (for stand alone operation). Other parameters allow the user to selectively disable control signals from the target system. This menu is also used for configuration of the memory map. The emulation memory is partitioned in 2K byte blocks. Each block of emulator memory can be designated as Read Only or Read/Write. Memory which is not mapped to the emulator is automatically mapped to the target system.

EVENT MENU:
There are 4 events (A,B,C and D) that are used for break and trigger operations. Each event is set as a bit pattern, including don't-care conditions of address, data, status (PSEN, INTA, RD, WR, OPFET, INT0, INTS, T0, T1, T2, T2EX) and 16 external trace lines. Each event has its own pass counter which can be programmed for 1-65535 counts.

Additionally, the address and data fields can be used to give range break and trigger conditions.

SEQUENCE MENU:
The four events can be logically combined to produce four independent actions:

1. Break emulation.
2. Trigger the trace buffer with a variable 0-65535 cycle delay.
3. Output an external logic level for triggering other instrumentation.
4. Measure the time interval from one event sequence to another. The time interval can be from 1 micro-second to 71 minutes.

The sequence can be from a pre-defined table or user defined. For example:

1. A then B then C then D
2. A then B without C
3. A or (B then C)

Additionally, the pulse output is configured in this menu to occur on one of the four events with either positive or negative true polarity. This output is typically used to trigger other instrumentation.

COMMAND MENU:
A powerful command set can be executed from this menu or from an alternate command line which some users may find faster to use than the descriptive menu format. The commands allow the user to begin emulation, single step, examine and modify memory and registers, in-line assemble and disassemble code, move memory, fill memory with a constant, search for a string in memory, test memory and read and write hex files in Intel, Tektronix or Motorola hex format.

TRACE MENU:
Real time trace information can be examined from this menu. The 8K X 88 bit trace is divided into 2 buffers which are 4K X 88 bits each. One trace buffer always provides a 4K X 88 bit trace history that terminates with the emulation break. The other trace buffer is triggered by the Trigger Point defined in the Sequence Menu with a 0-65535 delay count which can occur any time during emulation. The trace buffers are displayed as disassembled code and bus activity. An alternate display shows address, data, status and external trace bits.

INTERFACE MENU:
When interfaced with a host computer, the user has the capability of configuring commands in a Macro format. This allows a series of commands to be executed with one macro command.

INTERFACING

The HMI-200 series emulators can be connected to a terminal with an RS232 interface for stand alone operation or it can be slaved to any computer with an RS232 interface. Slave operation with a computer allows convenient transferring of hex files to and from the emulator and it also allows the emulator operation to be controlled from the computer console. The master RS232 port is used for control of the system while the auxiliary port can be used to download files from a separate source.

PACKAGING

The HMI-200 series emulator is packaged in a compact 12" X 9½" X 3½" case and weighs approximately 10 lbs. The emulation cable from the emulator to the 8051 socket is approximately 20" long with adaptors available for the DIP and PLCC packages.

POWER CONSUMPTION

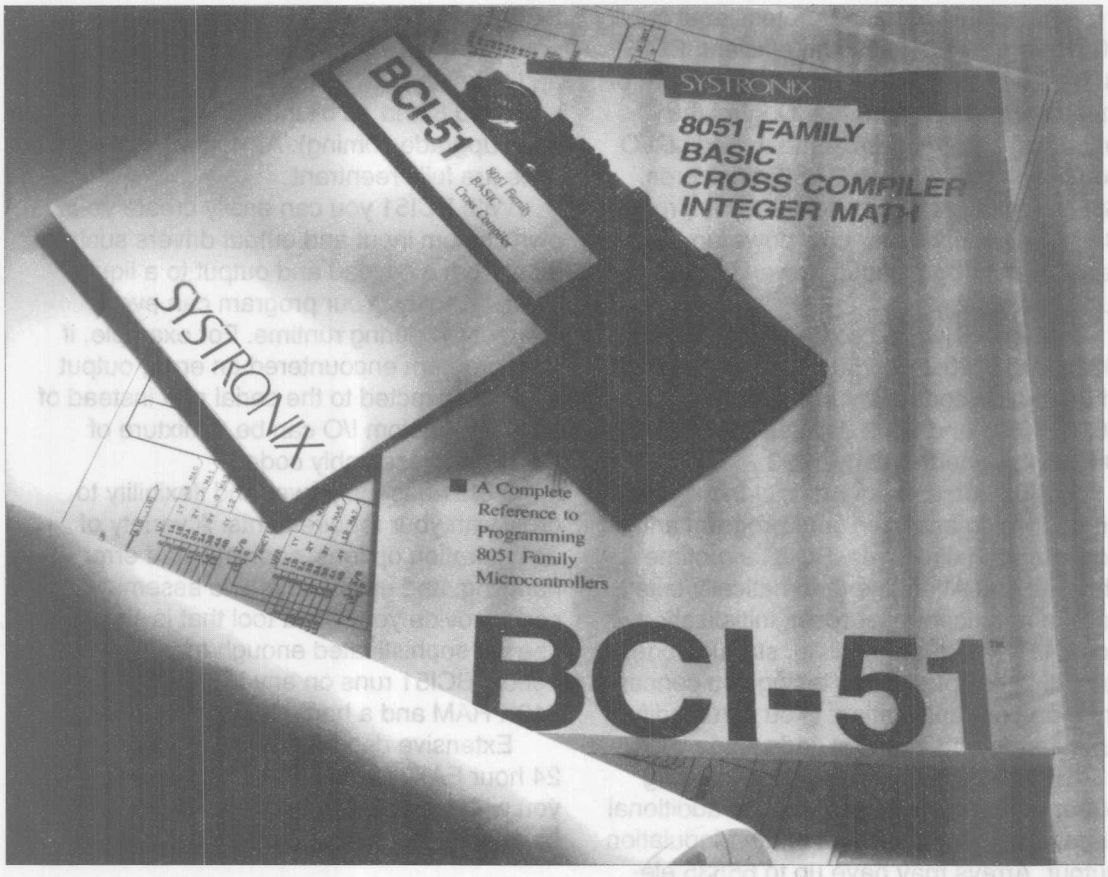
Switch selectable power supply accepts 110 VAC or 220 VAC at 75W.

Huntsville Microsystems, Inc.
P.O. Box 12415
4040 South Memorial Parkway
Huntsville, AL 35802
205-881-6005 TWX 510-600-8258
FAX 205-882-6701

Specifications subject to change without notice.

©Copyright Huntsville Microsystems, Inc. 1988

BCI51 Dallas Family BASIC Compiler



"[BCI51] makes a potent development combination with the DS5000TK."
***EDN*, January 20, 1992**

"[BCI51] has allowed me to program complex DS5000T projects that would have been unthinkable if faced with the burden of writing them in assembly code."
James Mason
DS5000T programmer.

Systronix

BCI51 Dallas Family BASIC Compiler

With BCI51, creating professional quality 8051 programs is easy and efficient. BCI51 is the only compiler with specific support for the unique Dallas features. Control the watchdog timer, clock/calendar, and all I/O ports, with easy-to-use BASIC keywords. Special reset/startup options let you provide BASIC subroutines for cold powerup, warm reset, watchdog timeout, powerfail, and runaway code.

Your code will be compatible with the Dallas DS5000, DS2250, and a wide range of 8-bit microcontrollers including 8031, 8032, 8051, and 8052 devices. DS5001 and DS2251 supported early 1993.

BCI51 contains an Artificial Intelligence engine which analyzes your program and configuration directives. BCI51's runtime libraries and AI engine automatically take care of microcontroller reset, initialization, serial I/O, interrupt handling, startup code, and all interrupt vectors, letting you concentrate on your application. (You can modify any startup and interrupt code if you wish.)

BCI51 supports interrupt-driven, ring buffered, full duplex serial I/O, an additional serial output, and a Pulse Width Modulation output. Arrays may have up to 65535 elements. Strings may be up to 255 characters long, and each may be a different length.

BCI51 includes a string concatenation operator to easily combine string variables and literal text. BCI51 provides signed and unsigned 8 and 16 bit integer math, (floating point upgrade coming). All numerical routines are fully reentrant.

With BCI51 you can easily create your own custom input and output drivers such as input from a keypad and output to a liquid crystal display. Your program can even redirect I/O during runtime. For example, if your program encountered an error, output could be directed to the serial port instead of an LCD. Custom I/O can be a mixture of BASIC and assembly code.

BCI51 has the power and flexibility to grow with your requirements. A variety of configuration options, sophisticated error handling, and unlimited in-line assembly code provide you with a tool that is simple to use yet sophisticated enough for all your needs. BCI51 runs on any MS-DOS PC with 512K RAM and a hard disk.

Extensive documentation, phone hotline, 24 hour FAX, and a 24 hour BBS provide you with outstanding support. All products have a 30-day money-back guarantee.

Also Available
- DS5000TK Development Kit
- BCI51 Assembly Language Programmer's Toolkit

Coming Soon
- DS2251 CPU Development Board
- DS2245/2271 Modem/Speech Development Board.

Systronix

555 South 300 East
Salt Lake City, UT 84111

TEL: (801) 534-1017
FAX: (801) 534-1019
BBS: (801) 487-2778

DS5000TK EVALUATION KIT

Introduction

The DS5000TK allows immediate evaluation of any DS5000 series device in an existing application. The kit comes supplied with DS5000T 32K-12 MHz, software diskette, and In-System Loader hardware. The kit supports in-system serial downloading of DS5000T from an IBM PC host. In addition, the DS5000T allows evaluation of all functions with the added feature of a real-time clock. Programs may be downloaded/verified from an Intel hex absolute object files residing on IBM PC. Demo routines are provided for evaluation of the timekeeping function. The software is supplied on a 5-1/4" floppy diskette.

Description

The In-System Loader hardware allows application software to be loaded into the DS5000 series device while it is connected to the target system, eliminating the need for removal of the device when reprogramming is required. KIT5K is a user-friendly PC software package which controls the loading process to the DS5000 while it is installed into the In-System Loader hardware. KIT5K provides a high-level user interface to the DS5000T via its Serial Load mode. When the program command is executed, the user is walked through a series of system configuration questions so that the DS5000T can be properly initialized before downloading takes place. Parameters such as the device's program/data memory mapping and software encryption operation are initialized in the proper order in this fashion. KIT5K manages all of the communication with the DS5000T during the downloading process so that the details of the serial download operation can remain transparent to the user.

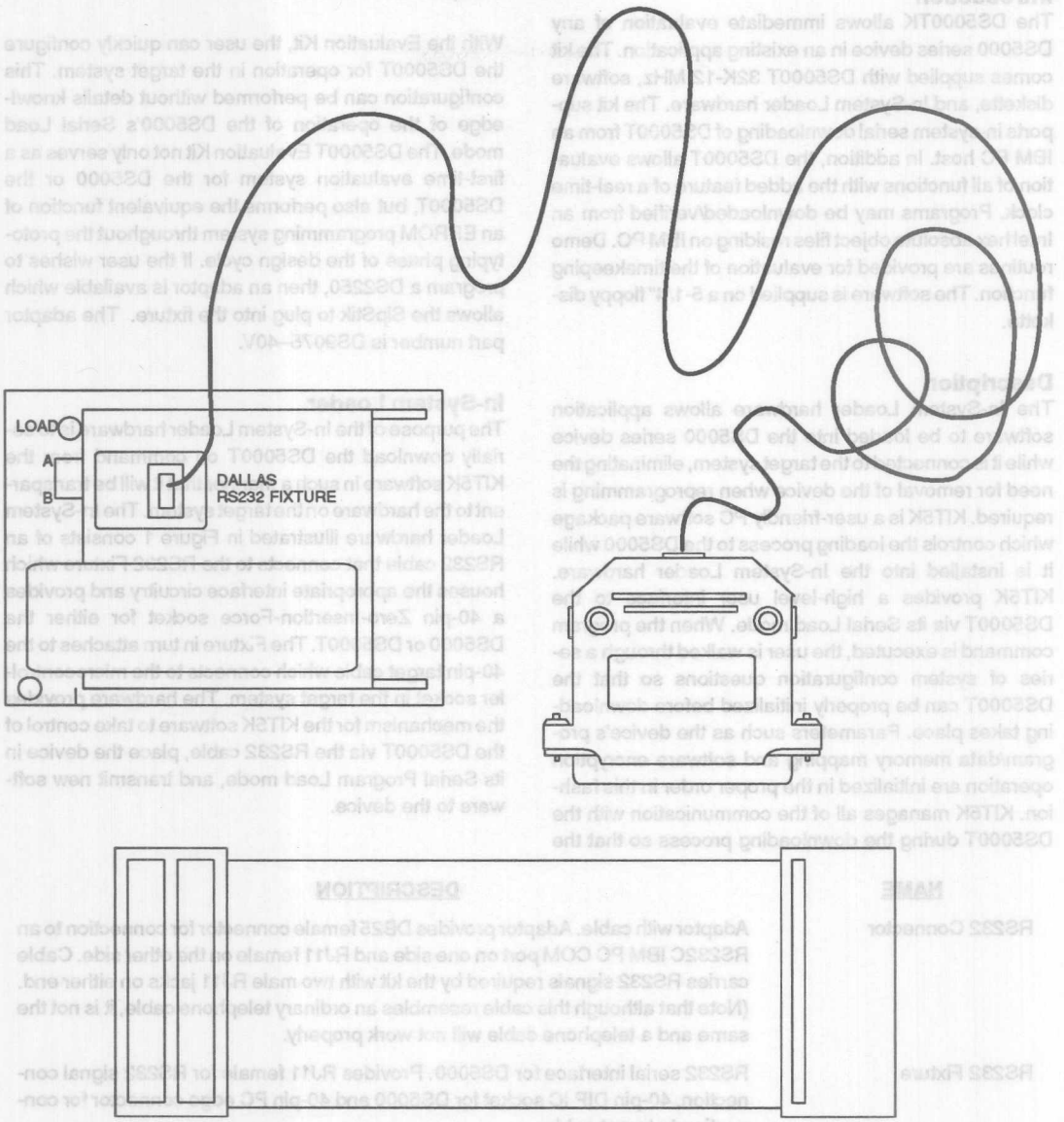
With the Evaluation Kit, the user can quickly configure the DS5000T for operation in the target system. This configuration can be performed without details knowledge of the operation of the DS5000's Serial Load mode. The DS5000T Evaluation Kit not only serves as a first-time evaluation system for the DS5000 or the DS5000T, but also performs the equivalent function of an EPROM programming system throughout the prototyping phase of the design cycle. If the user wishes to program a DS2250, then an adaptor is available which allows the SipStik to plug into the fixture. The adaptor part number is DS9075–40V.

In-System Loader

The purpose of the In-System Loader hardware is to serially download the DS5000T on command from the KIT5K software in such a manner that it will be transparent to the hardware on the target system. The In-System Loader hardware illustrated in Figure 1 consists of an RS232 cable that connects to the RS232 Fixture which houses the appropriate interface circuitry and provides a 40-pin Zero-Insertion-Force socket for either the DS5000 or DS5000T. The Fixture in turn attaches to the 40-pin target cable which connects to the microcontroller socket in the target system. The hardware provides the mechanism for the KIT5K software to take control of the DS5000T via the RS232 cable, place the device in its Serial Program Load mode, and transmit new software to the device.

NAME	DESCRIPTION
RS232 Connector	Adaptor with cable. Adaptor provides DB25 female connector for connection to an RS232C IBM PC COM port on one side and RJ11 female on the other side. Cable carries RS232 signals required by the kit with two male RJ11 jacks on either end. (Note that although this cable resembles an ordinary telephone cable, it is not the same and a telephone cable will not work properly.
RS232 Fixture	RS232 serial interface for DS5000. Provides RJ11 female for RS232 signal connection, 40-pin DIP IC socket for DS5000 and 40-pin PC edge connector for connection to target cable.
Target Cable	40-pin adaptor cable which connects the 40-pin edge connector on the RS232 Fixture to the target system microcontroller socket.

DS5000T EVALUATION KIT: IN-SYSTEM LOADER HARDWARE Figure 1



KIT5K OPERATION

KIT5K is the software environment supplied with the DS5000T Evaluation Kit. It provides a high-level interface for loading application software to the DS5000T or for setting its configuration parameters via the Program command. For more advanced users, KIT5K provides a number of commands which allow individual manipulation of the DS5000's resources. For example, these commands allow the direct initialization of the MCON register, loading of the 40-bit Encryption Key word, and setting and clearing of the Security Lock. In addition, an individual memory location examine and change capability is provided to allow patches to be made to the application software. When the loading operation is completed, the device can be released on command from the PC to run the application software.

For users interested in serial communication applications, versions of KIT5K (after 3.0) include a dumb terminal emulator. This code allows the user to run application software in the DS5000 which communicates with the user via a PC COM port. The serial communication uses the target system time base, allowing flexibility in baud rates. The terminal emulator simplifies applications which use the Dallas serial communication software module. Many of the KIT commands are available while in terminal emulation.

After KIT5K has been invoked and the prompt has been displayed (as described above), commands may be entered by the user. KIT5K operates either in interactive mode or in batch mode. The user will not see any of the communication between KIT5K and the DS5000T (except when debug is on). The following is a summary of commands recognized by the KIT5K software. Note, KIT5K will not work properly with DS5001 series parts unless used in terminal emulator mode. An upgrade software package is available on request.

KIT MODE

cd	Change to another directory or show the default directory.
cls	Clear screen.
com	Specify the COM port for the In-System Loader hardware.
dir	List the default directory or specified path.
display	List Embedded RAM contents in debug format.
do	Execute a list of KIT5K commands from a file.
dos	Enter DOS mode temporarily.
dtr	Toggles the DTR line to switch between run and load.
dump	Dump Embedded RAM in Intel Hex to a file.
edit	Individual examine/change Embedded RAM bytes in DS5000.
exit	Exit the KIT5K program; return to MS-DOS.
fill	Fill Embedded RAM with a constant value.
help	Describe the function and syntax of KIT5K commands.
key	Load the 40-bit Encryption Key word.
load	Load Embedded RAM from an Intel Hex object file.
lock	Set the Security Lock on the DS5000.
logoff	Disable logging of KIT5K commands.
logto	Log KIT5K commands to the specified file.
mcon	Set the MCON register with a specified value.
out1	Toggles a PC/AT bus signal for use by PC-based target systems.
partition	Set the Partition Address with a specified value.
pgmode	Place the in its Serial Load mode.
program	Program the DS5000 automatically with a configuration file.
quit	Same as exit; leave the KIT5K program; return to MS-DOS.
range	Set the Range Address to 8K or 32K.
reload	Sets the PC UART reload value for flexible baud rate selection.
run	Specify the serial baud rate to be used during loading.
speed	Specify the serial baud rate to be used during loading.
status	Display status.
term	Switch to terminal mode; see below.

type	Type the requested filename to the screen.
unasm	Unassemble the program memory to view code.
unlock	Clear Security Lock.
verify	Verify Embedded RAM with the specified Intel Hex file.

TERMINAL MODE

capture	Capture RAM to memory and allow uploading to a file.
cd	Change directory.
cls	Clear screen.
com	Set the com port for communication.
dir	Read directory.
dos	Enter DOS mode temporarily.
dtr	Toggle the DTR line.
exit	Exit Kit.
help	Describe the functions and syntax of terminal emulator.
kit	Switch to kit mode.
logoff	Stop logging commands to a file.
logto	Log commands to a file.
nosnow	Enable snow checking for CGA screens.
out1	Toggles a PC/AT bus signal for use by PC-based target systems.
pgmode	Program the DS5000 automatically from a configuration file.
quit	Leave KIT5K.
reload	Sets the PC UART reload value for flexible baud rate selection.
run	Run a DS5000 program.
send	Send a file to the DS5000 via the serial bus.
snow	Disable snow checking for CGA monitors.
speed	Select the baud rate.
type	List a DOS file.

System Requirements

The Evaluation Kit requires an IBM PC or compatible with DOS 2.0 or later and at least 128K bytes of memory. In addition, an RS232 port must be available which is configured as COM1 (03F8H, IRQ4) or COM2 (02F8H, IRQ3). Displays which are supported include monochrome, color graphics (CGA), or enhanced graphics (EGA mode 3).

Power (+5V) must be supplied to the RS232 fixture from the V_{CC} pin of the target system via the Target Cable (see below). A user selected crystal is needed to run programs.

Electrical Specifications

Operating Temperature Range	0 to +50° C
System Power Supply Requirements	
from Target System	+5V @100mA max
(DS5000) installed in target system;	
no load on port pins, $\overline{PSEN}$, ALE	50 mA typical

Interface

Connectors:
25-pin RS232 'D' type to RJ11 jack adaptor
RJ11 on RS232 Fixture
40-pin card edge (0.1" centers) on RS232 Fixture
40-pin edge connector on Target Cable
40-pin DIP plug on Target Cable

Interfacing to an IBM PC COM Port

The DS5000TK Evaluation Kit is provided with serial interface cables which are designed to be connected directly to a standard male DB25 connector from a COM port on an IBM PC. However, in some PC configurations, the DB25 connector may not be supplied from the PC and/or the pin assignments may be different. Consequently, problems may be experienced by users in trying to establish communication between the PC and these kit products.

In order to solve this problem, the following documentation is provided to define the RS232 pin assignments

between the PC and RJ11 jack which is present on the DS5000TK hardware module. If necessary, the user can then build an adaptor to insure that the appropriate signals from the PC are routed to the correct pins on the RJ11 jack connector for the DS5000TK.

Configuration

The DS5000TK is configured as a DCE RS232 interface. A total of four signals are routed from the PC's COM port connector to the kit hardware as summarized below:

IBM PC SIGNAL	DB25 PIN #		KIT HARDWARE
TXD	Pin 2	→	RXD
RXD	Pin 3	←	TXD
GND	Pin 7	—	GND
DTR	Pin 20	→	DTR

The DB25 to RJ11 adaptor takes the above four signals from the IBM PC's COM port and routes them to its RJ11 jack.

RJ11 Cable

The kits are supplied with an interconnecting cable with two RJ11 plugs on either end. This cable is used to route the above RS232 signals from the DB25/RJ11 adaptor to the RJ11 jack located on the kit hardware. Although this cable appears to be a standard telephone cable commonly used in the United States, it is a special version which has pin assignments which are reversed from the U.S. standard cable. As a result, such a telephone cable will cause the kits to not operate.

Looking at the end of the RJ11 cable as a reference, the pin assignment for the cable from left to right can be given as follows:

PIN #	→ 1	2	3	4
Wire Color	← Black	Red	Green	Yellow
DCE Signal	→ RXD	GND	DTR	TXD

Kit Hardware RJ11 Jack

In the DS5000TK there is an RJ11 jack which is mounted on the hardware fixture which accepts the cable described above. When the cable is installed in the jack, there are leads exposed on the jack which are electrically connected to signals from the cable. As a result, these lines may be probed if necessary to insure that the appropriate RS232 signals from the PC COM port are correctly routed to kit hardware.

DS5000TK Software Diskette

The software diskette supplied with the DS5000TK includes KIT5K as well as a set of programs which word together to allow you to read and set the date and time in a DS5000T and to compare it with the clock in your IBM or IBM-compatible PC. All of these programs are summarized below:

1. KIT5K.EXE — This is the utility program which is used for programming the DS5000T. It can be used to read programs to and from the DS5000T, examine the contents of program memory, set the range of 8K or 32K, and set or clear the program memory encryption key.

2. TEST.HEX — This is a short file of text which can be used to confirm that the KIT5K program can load data in to the DS5000T successfully.

3. DEMO.BAT — This is a batch file which automatically loads the DS5000T with a program to read the time and then runs a program in the PC to set or display the time sent back by the DS5000T.

4. DEMOS5T.CMD — This is the command file which directs the KIT5K program to load the DEMODS5T.HEX program into the DS5000T.

5. DEMODS5T.HEX — This DS5000T program responds to commands received over the serial I/O port to send or receive the date/time information between the embedded clock/calendar in the DS5000T and the serial I/O port. This allows an external computer to access to the date/time information.

6. DEMODS5T.LST — This is a listing of the program DEMODS5T.HEX described above.

7. DEMODS5T.SRC — This is the source code for the program DEMODS5T.HEX described above.

8. DEMODS5T.EXE — This is a PC program to test the performance of a DS5000T by relaying date/time information between it and the PC and comparing it against the PC clock. This program requires that DEMODS5T.HEX be installed in the DS5000T, and that the DS5000T be installed in a RS232 Fixture with its serial selection switch in the B position and connected to one of the COM ports of the PC. The target system must have an 11.0592 MHz crystal.

9. DEMODS5T.PAS — This is the Pascal source code for the program DEMODS5T.EXE described above.

10. SAMPLE.HEX — This program responds to commands received over the serial port to set the date and time information in the DS5000T.

11. SAMPLE.LST — This is the listing of the program SAMPLE.HEX described above.

12. SAMPLE.SRC — This is the source code for the program SAMPLE.HEX described above.

Installation Instructions

To install the In-System Loader hardware, the user should first turn power off to the target system. Next, the RS232 Cable with the 25-pin DB25 adaptor should be connected to either the COM1 or COM2 port of the IBM PC. Note that although this cable resembles an ordinary telephone cable, it is not the same and a telephone cable will not work properly. The RJ11 plug should then be inserted into the RJ11 jack on the RS232 Fixture. Finally, the 40-pin Target Cable should be connected between the edge connector of the RS232 Fixture and the 40-pin socket of the target system. The user should check to insure that the switch on the RS232 Fixture is in the "A" position for initial checkout.

The KIT5K software is shipped on a 5-1/4": flexible diskette. As a precaution, a duplicate copy of the diskette should be made before attempting to run the software on the PC. The software may be executed directly from diskette, or it can be copied over to a hard disk drive and executed from there.

Initial Checkout

KIT5K is supplied with a file called "TEST.HEX" that may be downloaded to the DS5000T to verify that the Evaluation Kit has been installed correctly. Below is a checkout procedure which downloads the "TEST.HEX" file to the DS5000T and initializes the device's configuration parameters. This procedure also demonstrates the command sequence which is likely to be the one most often used to serially download new application software.

After performing the installation procedure described above, the initial checkout procedure can be started by executing the KIT5K software. Execution of KIT5K can be invoked by typing

>kit5K <CR>

following the prompt. The software will respond with the following:

Dallas Semiconductor DS5000 Evaluation Kit Software
Kit5K – Version x.x
Copyright (C) 1987, Dallas Semiconductor Corporation

kit5K>

At this point, KIT5K is ready to accept commands. The first action which should be taken is to specify the COM port to which the In-System Loader hardware is connected. Assuming that this is COM1, the user would enter the following:

kit5K>com 1 <CR>

KIT5K may now communicate with the DS5000's on-chip serial loader. In order to download the file 'TEST.HEX' to the DS5000, the user should type:

```
kit5K>program test.hex <CR>
```

The user will then be prompted with a series of questions regarding the configuration of the DS5000. By entering a <CR> character for all of these, the displayed default values will be assigned as shown below:

```
Configuration: Press return for default value
    Range (8000) = <CR>
    Partition Address (1000) = <CR>
    Encryption (No) <CR>
    Locking (No) = <CR>
    Begin (0000) = <CR>
    End (7FFF) = <CR>
    Verify (Yes) = <CR>
```

After this sequence has been completed, the software will echo back the selected configuration and program the DS5000 series device as follows:

```
Configuration Contents:
    Range: 8000
    Partition Address: 1000
    Encryption: No
    Locking: No
    Begin: No
    End: 7FFF
    Verify: Yes

    Unlocking...
    Clearing MCON...
    Setting Range...
    Setting Partition
    Loading...
    Verifying...
    kit5K>
```

The above response from KIT5K signifies that the configuration/loading operation was completed successfully. The user may verify this by executing the command "DISP 0 80" to show the test data in the DS5000T memory. This is shown below:

```
kit5K>disp 0 80 <CR>
0000 01 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00  ................
0010 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00  ................
0020 54 45 58 54 20 46 4F 52 - 20 44 49 53 50 4C 41 59  TEXT FOR DISPLAY
0030 20 50 55 52 50 4F 53 45 - 53 2E 20 41 42 43 44 45   PURPOSES.ABCDE
0040 46 47 48 49 4A 4B 4C 4D - 4E 4F 50 51 52 53 54 55  FGHIJKLMNOPQRSTU
0050 56 57 58 59 5A 20 61 62 - 63 64 65 66 67 68 69 6A  VWXYZ abcdefghij
0060 6B 6C 6D 6E 6F 70 71 72 - 73 74 75 76 77 78 79 7A  klmnopqrstuvwxyz
0070 20 31 32 33 34 35 36 37 - 38 39 30 20 30 98 FD C2   1234567890 0...
0080 98                                                 ...
```

Following the execution of the Program command shown above, KIT5K automatically created a configuration file with the same name as the specified .HEX file, but with an extension of .CFG. This file is used to save the DS5000 configuration parameters that the user specified during the first time that the Program command was executed for the specified .HEX file. During future Program command operations, the associated configuration file will be used to initialize the parameters automatically without any intervention by the user. If modification of the previously selected parameters is desired, the user should select the /edit when invoking the Program command as shown below:

```
kit5K>program test.hex /edit<CR>
```

Command Line Syntax

Single-letter ASCII character strings are recognized as commands. Commands will not be processed until an entire command line is entered and terminated with a <CR>. Since a command line is not processed until a <CR> is entered, it may be edited with the delete key which will do a destructive delete to the screen.

Only legal characters will be echoed back to the screen. The legal characters are: 0123456789<:>, <space>, ABCDEFGHIJKLMNOPQRSTUVWXYZ, abcdefghijklmnopqrstuvwxyz, and <delete>. Backspace characters (<BS>) are converted to delete characters. The horizontal tab character is converted to space.

In most commands, arguments are represented by hexadecimal numbers. A hexadecimal number is any sequence of hexadecimal characters. A hexadecimal character may be a digit, 0 through 9, or one of the letters A through F. A byte will always be the right-most two digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following bytes:

A	→	0AH
AB	→	0ABH
ABC	→	0BCH
ABCD	→	0CDH

An address will always be the right-most four digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following addresses:

A	→	000AH
AB	→	00ABH
ABC	→	0ABCH
ABCD	→	0ABCDH
ABCDE	→	0BCDEH

In the other cases, arguments are represented by decimal numbers. In commands which normally accept hexadecimal numbers as their arguments, decimal operation may be invoked by typing a period (.) after the entered value.

KIT5K Detailed Command Summary (Kit Mode)
Cd directory_name
The default directory is set to the argument.

Example:

```
kit5K> cd \ws
 C:\WS

kit5K>
```

Cls
Clear the monitor screen

Com (1) (2)
Selects which PC communication port to use. DOS puts the port base numbers into a vector area in memory. This program will attempt to use either of these two addresses (there may be zero or one value here) after the base has been checked against the hexadecimal base values of 3F8 and 2F8.

Example:

```
kit5K>com1

kit5K>
```

Dir (filename | wildcards)
A list of the file names in the directory are displayed.

Example:

```
kit5K> dir
RUNME.EXE READ.ME KIT5K.EXE TEST.HEX DEMO.BAT DEMODS5T.CMD DEMODS5T.EXE DEMODS5T.HEX
DEMOS5T.LST DEMODS5T.PAS DEMODS5T.SRC SAMPLE.HEX SAMPLE.LST SAMPLE.SRC
kit5K>
```

Display beg–addr, end–addr (>filename)
Data is displayed as in MS-DOS debug. The optional arguments are interpreted as hexadecimal numbers as a default. The beginning and ending are positional and will default to zero and the current range value, if not specified. The "greater than" sign signifies that output is to go to the specified file instead of to the screen.

```
kit5K>disp0 80 <CR>

0000 01 00 00 00 00 00 00 00 – 00 00 00 00 00 00 00 00 ................
0010 00 00 00 00 00 00 00 00 – 00 00 00 00 00 00 00 00 ................
0020 54 45 58 54 20 46 4F 52 – 20 44 49 53 50 4C 41 59 TEST FOR DISPLAY
0030 20 50 55 52 50 4F 53 45 – 53 2E 20 41 42 43 44 45  PURPOSES. ABCDE
0040 46 47 48 49 4A 4B 4C 4D – 4E 4F 50 51 52 53 54 55 FGHIJKLMNOPQRSTU
0050 56 57 58 59 5A 20 61 62 – 63 64 65 66 67 68 69 6A VWXYZ abcedfghij
0060 6B 6C 6D 6E 6F 70 71 72 – 73 74 76 77 78 79 7A    klmnopqrstuvwxyz
0070 20 31 32 33 34 35 36 37 – 38 39 30 20 30 98 FD C2  1234567890 0.....
0080 98

kit5K>
```

Do filename
The commands in the filename are executed as a script until the end of file is encountered or until an error occurs.

Example:

```
kit5K>type script
fill ff 0 2f
display 0 2f

kit5K> do script

fill ff 0 2f

display 0 2f
0000 FF FF FF FF FF FF FF FF – FF FF FF FF FF FF FF FF ................
0010 FF FF FF FF FF FF FF FF – FF FF FF FF FF FF FF FF ................
0020 FF FF FF FF FF FF FF FF – FF FF FF FF FF FF FF FF ................

kit5K>
```

Dos
Takes the user to the DOS prompt to execute commands without actually leaving KIT5K. When DOS commands are complete, return to KIT by typing "exit".

Example

```
kit5k>dos
Type Exit to return to Kit5K

DOS Message

c:\kit>copy \test.hex \kit\prog.hex

c:\kit>exit

Welcome back to KIT
```

Dtr
Toggles the status of the DTR line. This will toggle between run and program load.
Example:

```
kit5k>dtr
dtr dropped on COM 1

kit5k>dtr
dtr asserted on COM 1
```

Dump beg-addr, end-addr (> filename)
Data is displayed as Intel Hex data. The optional arguments are interpreted as hexadecimal numbers as a default. The beginning and ending addresses are positional and will default to zero and the current range value if not specified. The "greater than" sign signifies that output is to go to the specified file instead of to the screen.

Example:

The following command dumps the specified range of data to the screen in Intel Hex formal:

```
kit5K> du 0 2f
:200000000010203FF05FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF0
:10002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFE0
:00000001FF

kit5K>
```

Alternatively, the command sequence shown below will dump the same data to a file called TEMP.HEX

```
kit5K> du 0 2f > temp.hex

kit5K>
```

Edit address
Data beginning with this address is available for examination and change. The argument is interpreted as a hexadecimal number as a default. In this example, user typed values are in italics.

Example:

```
kit5K> edit 0
        X to exit edit; CR to leave unchanged; NUMBER to change.
        0000            FF:00
        0001            FF:01
        0002            FF:02
        0003            FF:<or>
        0004            FF:04
        0005            FF:x


kit5K> display 0 f
0000 00 01 02 FF 04 FF FF FF - FF FF FF FF FF FF FF FF .................

kit5K>
```

Exit
Buffers and files are closed, the communication ports are cleared and the program is exited.

Example:

```
kit5K>exit

A:\>
```

Fill value (beg-addr, end-addr)
The specified range of locations is filled with the specified constant value. The optional arguments are interpreted as hexadecimal numbers as a default.

Example:

```
kit5K>fill ff 0 2f

kit5K>display 0 2f

0000 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................
0010 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................
0020 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................

kit5K>
```

Help (command-name)
A list of available commands is listed or hep on the requested command is given.

Example:

```
kit5K?help load
        name: load
    function: Load the DS5000 memory from a file containing Intel Hex data.

kit5K>
```

Key (ASCII String | 5 pairs of hex digits | RANDOM)
This command sets the Encryption Key Word within the DS5000. The key may be set in three different ways. Five ASCII characters enclosed in single or double quotes may be used to specify the key. The user may request the computer to specify five random numbers or the user may enter five hexadecimal numbers; e.g. KEY 'Stuff', or KEY AABBCCDDEE, or KEY RANDOM.

Example:

```
kit5K>key 001223344

kit5K>
```

Load filename
The data from the filename is used to program the chip. Arguments are interpreted as hexadecimal numbers as a default. Several errors can be detected, such as multiple programming of the same memory location and attempts to program nonexistent memory. All error messages will be reported. No configuration data files are used with this load. See the description of the Program command.

Example:

```
kit5K>load test.hex

kit5K>display 0 7f
```

```
0000 01 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00 ................
0010 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00 ................
0020 54 45 58 54 20 46 4F 52 - 20 44 49 53 50 4C 41 59 TEST FOR DISPLAY
0030 20 50 55 52 50 4F 53 45 - 53 2E 20 41 42 43 44 45  PURPOSES. ABCDE
0040 46 47 48 49 4A 4B 4C 4D - 4E 4F 50 51 52 53 54 55 FGHIJKLMNOPQRSTU
0050 56 57 58 59 5A 20 61 62 - 63 64 65 66 67 68 69 6A VWXYZ abcedfghij
0060 6B 6C 6D 6E 6F 70 71 72 - 73 74 75 76 77 78 79 7A klmnopqrstuvwxyz
0070 20 31 32 33 34 35 36 37 - 38 39 30 20 59 F6 41 11  1234567890 Y.A.
```

kit5k>

Lock
Set the Security Lock on the DS5000.

Example:

kit5K>lock

kit5K>status
    Part is locked.
kit5K>

Logoff
The data logging buffers are flushed and the log file is closed.

Example:

(see the example given below for the Logto command)

Logto filename
Data logging is enabled to the specified file. Logging is turned off with the Logoff command or by exiting. All user input and data echoed to the screen is logged.

Example:

kit5K>logto script

kit5K>fill ff 0 2f

kit5K>display 0 2f

```
0000 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................
0010 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................
0020 FF FF FF FF FF FF FF FF - FF FF FF FF FF FF FF FF ................
```

kit5K>logoff

kit5K>

MCON number
The MCON register is changed to the requested number. The argument is interpreted as a hexadecimal number as a default.

Example:

kit5K>mcon 28

```
kit5K>status
        Partition: 1000  Range: 8000 (32K)  Chip Enable 2: 0
kit5K>
```

Partition (0 | 800 | 1000 | 1800 | 2000 | 2800 | 3000 | 3800 | 4000 | 4800 | 5000 | 5800 | 6000 | 6800 | 7000 | 8000)
The Partition Address is set to the selected hexadecimal value and the display is updated. The argument is interpreted as a decimal number as a default.

Example:

```
kit5K>par 2000
kit5K>status
        Partition: 2000  Range: 8000 (32K)  Chip Enable 2: 0
kit5K
```

Out1
Toggles the status of the out1 signal on the PC/AT bus. This is useful for PC-based target system. It may serve in a similar manner to DTR.

Partition
Directly sets the partition as above.

Pgmode
Signals the In-System Loader hardware to place the DS5000 in Program mode by raising the DTR line to its active level. Communication is then attempted to the DS5000 at the selected baud rate. The default value of the baud rate is 9600 bps if no other baud rate frequency has been selected.

Example:

```
kit5K>pgmode
        DTR asserted on COM 1
kit5K>
```

Program filename [/edit]
The data from the filename (default extension = .HEX) and its associated configuration file (.CFG) is used to program the chip. If the configuration file dows not exist, the necessary information will be prompted for and a new configuration file will be created. If the file doesn't exist during batch operation, defaults will be used. The configuration file is displayed before the part is programmed. The /Edit switch allows the current configuration to be edited one line at a time.
Example:

(see Initial Checkout section)

Range (2000 | 8000)
The range bit is set and the display is updated.

Example:

```
kit5K> range 8000

kit5K>status
        Partition: 2000 Range: 8000 (32K) Chip Enable 2: 0
kit5K
```

Reload
Sets the reload value of the PC 8250 UART. This is available for more flexible selection of baud rates. The baud rate is selected by the formula 115200/n where n is the reload number.

Example:
kit5K>reload 6

baud rate 19200

Run
The DTR line is dropped to its inactive level. This action signals the In-System Loader hardware to release the DS5000 from its Program mode and allows it to run the application program. A crystal must be connected to the target system.

Example:

kit5K>run
DTR dropped from COM 1
kit5K>

Speed (150 | 300 | 600 | 1200 | 2400 | 4800 | 19200)
This command is used to select the baud rate for communication between the PC and the DS5000. Decimal numbers are used in entering the arguments.

Example:
kit5K> sp 1200

kit5K>

Status
Display the current status of the DS5000.

Example:

kit5K>status
Partition: 2000 Range: 8000 (32K) Chip Enable 2: 0
kit5K>

Term
Switch to terminal emulation mode. This allows testing of software which expects serial commands from a PC or similar station. Many of the kit commands are available from this mode. Online help is also available as with normal KIT mode.

Example:

kit5K> term

———— Dumb Terminal Press Escape for Commands ————

–
"press escape"
–
dt> kit

kit5k>

Type filename
Type the requested file to the screen.

Example:

```
kit5K> type script
fill ff 0 2f
display 0 2f

kit5K>
```

Unasm
Unassemble the program RAM of the DS5000. Optional specify start and stop addresses for disassembly.

Example:

```
kit5k> usasm 100 10C

0100    EB      MOV     A,R3
0101    F7      MOV     @R1,A
0102    5191    ACALL   00291H
0104    B40A07  CJNE    A,#00AH,0010EH
0107    311F    ACALL   0011BH
0109    20050F  JB      005H,0011BH
010C    01DB    AJMP    000DBH

KIT5K>
```

Unlock
Clear the Security Lock bit on the DS5000.

Example:

```
kit5K> unlock

kit5K>
```

Verify Filename (beg-addr, end-addr)
The data from the filename is used to verify data in the chip. The optional beginning and ending addresses will determine what parts of the input file will be used for verification. Data from the file is compared against the current contents of memory. Anytime they are different, the discrepancy will be reported. Arguments are entered as hexadecimal numbers. The beginning and ending addresses are positional and will default to zero and to the current range value if not specified. No configuration data files are used with this load.

Example:

```
kit5K>load test.hex

kit5K>verify test.hex
      No Verification errors
kit5K>
```

Terminal Emulator Mode Commands

This section discusses the commands which are unique or different in the terminal mode. A complete list of commands for this mode is given above.

Capture
When toggled on, the capture command receives data from the DS5000 and holds the contents into memory. When toggled off, the program prompts for a filename to store to.

Example:

```
dt> capture
      "an operation which sends data to the COM port"
      "escape"

dt>capture
Enter filename to save capture buffer, CR to discard.
      Filename:
```

Help
Displays the commands available in terminal mode

Example:

```
dt> help
   Available commands:
      CApture     CD      CLs     COm     DIR     DOS
      DTr         EXit    Help    Kit     LOGOff  LOGTo
      NosnowOut1  PGmode  Quit    REload  RUn
      SEnd        SNow    SPeed   TYpe
   Use "Help" command-name" or HELP*" for more information.
```

KIt
Switch out of terminal emulator into KIT.

Nosnow
Enable snow checking for CGA monitors. Screen output is slower.

Send
Sends a file through a COM port. Useful for serial download to an application program.

Snow
Disable snow checking on CGA tubes. Output is faster.

In-System Loader Operational Details

The DTR signal on the COM port interface is activated by the KIT5K software to signal the hardware in the RS232 Fixture that serial downloading is in effect. In addition, RTS is activated for the duration of the time that KIT5K is in execution.

When DTR is activated, the hardware isolates signals on the DS5000 which are used to accomplish the serial download task from the target system circuitry. This includes RST, TXD(P3.1), RXD (P3.0), and XTAL1. RST is then driven high to initiate a reset within the DS5000. Following this action, XTAL1 is then driven with the RS232 Fixture's 11.0592 MHz clock oscillator circuit. Finally, $\overline{\text{PSEN}}$ is then driven low. This sequence of actions causes the DS5000 to begin operation in its Serial Program Load mode at a clock frequency from which standard baud rate frequencies may be derived. None of the activity on the RST, TXD, RXD, and XTAL1 pins is driven out to the target system lines while the DTR signal is active. DTR will remain active until either the "Run" or the "Exit" command is executed from KIT5K.

When DTR is released, the In-System Loader causes the DS5000 to be reset and begin execution of the application code. When this occurs and the In-System Loader's switch is in the "A" position, all communication performed on the TXD and RXD pins on the target system is isolated from the PC. If the switch is placed into the "B" position, then the TXD and RXD pins are con-

nected to the PC and not to the target system during the execution of the application program. This capability is provided for applications which require that the target system have the ability to communicate to the PC through the COM port used by KIT5K. In general, however, the "Run" command cannot be used to initiate the execution of DS5000 application software which communicates to the IBM PC since KIT5K will still be in execution and will have control of the COM port.

When operating the serial loader, the user should insure that the port pins P2.7 and P2.6 are not externally pulled to a low state. If this condition were to occur, the Parallel Loader mode's verify cycle would be invoked. As a result, serial communications between the KIT5K program on the PC and the serial loader on the DS5000 would not function. This is due to the fact that since RST and $\overline{PSEN}$ are being pulled high and low respectively to invoke the DS5000's Serial Load mode operation, the addition of P2.7 and P2.6 being pulled low would satisfy the condition for a Verify cycle on the device. The easiest way to insure that this does not occur is to assign either P2.7 or P2.6 as an output in the target system. This will insure that at least one of these pins will be pulled high by its internal pullup while the Serial Load mode is in effect, as all port pins are set at a 1 (reset) state during this time.

KIT5K OPERATIONAL DETAILS

When KIT5K is invoked from MS-DOS, communication is automatically established with the DS5000 via its on-chip Serial Loader. KIT5K always configures the COM port with eight data bits, one stop bit and no parity, and activates RTS. Communication with the DS5000 is then attempted by asserting DTR and sending a carriage return (CR>) at the currently selected baud rate (default value is 9600 bps). If no response is detected, then an error message to this effect is displayed to the user.

After the baud rate is determined, KIT5K holds both the DTR line and the RTS line at their active levels until either a "Run" or and "Exit" command is executed. When the "Run" command is executed, DTR is taken inactive. When the "Exit" command is issued, both DTR and RTS are de-asserted and control is returned to the MS-DOS operating system.

DS5000T Demonstration Software

The DS5000TK diskette contains a set of programs which word together to allow you to read and set the date and time in a DS5000T and to compare it with the clock in your IBM or IBM-compatible PC. The program DEMODS5T.HEX runs in the DS5000T, and is used to pass date and time information back and forth between the DS5000T and the PC over the serial port. The program DEMODS5T.EXE runs in the PC and allows you to read or set the time in the DS5000T and compare it with the PC clock.

Running DEMO.BAT

DEMO.BAT automatically loads the DS5000T with a program (DEMODS5T.HEX) to read the time and then runs a program in the PC (DEMODS5T.EXE) to set or display the time sent back by the DS5000. The following steps are required to run this demo:

a) Place your DS5000T in the zero insertion force socket of an RS232 Fixture and turn the socket lock screw with a small screwdriver to the "C" position.

b) Place the switch on the fixture in the position labeled B, then connect the fixture using the ribbon cable to a socket providing V_{CC} and Ground. The socket must also supply an 11.0592 MHz crystal connected across pins 18 and 19, with pins 18 and 19 also each connected through a 33 pF capacitor to pin 20 (V_{SS}). Finally, the socket should insure that pin 31 (EA) is tied to pin 40 (V_{CC}).

c) Connect the Fixture to the COM1 serial port of your PC using the modular cable provided. (Note that although this cable resembles an ordinary telephone cable, it is not the same, and a telephone cable will not work properly.)

d) If your PC does not have a built-in clock, use the DATE and TIME commands in DOS to set the correct date and time in your computer.

e) Type DEMO on the keyboard to begin execution of the batch file. This batch file first uses the KIT5K utility program to load a program into the DS5000T which relays the date and time information back to the PC, then it runs a program in the PC which compares the time provided by the PC clock with the time provided by the DS5000T.

f) Within a few seconds, the program should report that it has found the DS5000T and it will ask you if you want to set the time. If you answer N, the program will begin reading the date and time information from the DS5000T and comparing it with the date and time from the PC clock. If you answer Y, the program will set the date and time in the DS5000T to agree with the PC clock, then it will begin comparing the two times. The program will continue running until any key is pressed.

DEMODST.HEX Operation

DEMODS5T.HEX is the program that response to commands received over the serial I/O port to send or receive the date/time information between the clock/calendar in the DS5000T and the serial I/O port. This allows an external program access to the date/time information. DEMODS5T.LST is a assembly language listing of this program.

The program first sets up the serial port for transmission at 9600 baud with eight data bits, no parity, and one stop bit. Next, the program begins execution of a loop waiting for an instruction from the serial port. Two valid instructions, R and W, are recognized. Receipt of the R character causes the DEMODS5T program to read eight bytes of date/time information from the embedded clock/calendar and send them out over the serial port. Receipt of the W character causes the DEMODS5T program to wait for eight bytes of date/time information from the serial port and write them to the embedded clock/calendar. Any other byte received from the serial port is incremented and then set back out to the serial port.

SAMPLE.HEX Operation

SAMPLE.HEX is a program which responds to commands received over the serial port to set the date and time information in the DS5000T. The program first initializes the serial port for communication at 9600 baud with eight data bit, no parity, and one stop bit. After setting the date and time, the program begins execution of an infinite loop which sends back the date and time each time a character is received. SAMPLE.SRC is the source code for the program SAMPLE.HEX. SAMPLE.LST is the assembly language listing of the program SAMPLE.HEX.

INDEX

INDEX

Numbers

128K X 8 SRAM, 3, 10, 16, 19, 58
32K X 8 SRAM, 2, 10, 13, 16, 17, 51
8K X 8 SRAM, 17

A

Address Enable (AE), 20, 22, 25, 42, 67, 68, 105
Addressing Modes, 45
Application, 1, 8, 12, 13, 15, 16, 17, 19, 20, 21, 22, 23, 25, 37, 39, 40, 43, 44, 59, 60, 61, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 76, 79, 80, 81, 83, 85, 86, 87, 88, 97, 103, 109, 113, 114, 115, 117, 118, 121, 122, 123, 124, 134, 140, 145, 146, 147, 157, 161, 166, 168
Assemblers, 45, 134, 141
Auto–Baud Rate Detection, 135, 136

B

Battery, 1, 49, 56, 57, 58, 59, 166
Battery Attach, 56, 57
Baud Rate, 30, 110, 111, 112, 113, 114, 117, 118, 121, 122, 123, 124, 132, 134, 135, 136, 137, 145
Block Diagram, 6, 7, 49, 50, 53, 54, 55, 68, 75, 108, 109, 114, 116, 117, 118, 119, 120, 146, 153
Boostrap Loader, Commands, 137
Bootstrap Loader, 1, 3, 8, 12, 14, 15, 20, 22, 37, 39, 40, 42, 59, 67, 70, 71, 74, 77, 79, 80, 131, 134, 135, 137, 138, 144, 145, 167
Bootstrap Loader, Invoking, 131, 132
Brownout, 64
Bus Organization, 6
Byte–wide Bus, 1, 2, 3, 8, 12, 13, 14, 16, 17, 18, 19, 23, 38, 39, 40, 49, 74, 76, 79, 80, 146
Byte–wide Memory, 6, 8, 147

C

Calendar, 146, 152, 155, 158
Clock, 1, 4, 9, 34, 61, 62, 64, 66, 68, 85, 86, 95, 99, 106, 110, 111, 112, 113, 114, 117, 118, 121, 122, 125, 126, 128, 130, 132, 140, 144, 146, 147, 149, 150, 151, 152, 157, 158, 161, 166
Code Example, 72, 88
Command Summaries, 138
Compilers, 11, 45, 167
Counters, 2, 6, 8, 33, 61, 68, 96, 97, 106, 107, 108, 109, 110, 113, 117, 124, 126
CRC, 3, 37, 66, 70, 71, 72, 77, 81, 84, 131, 137, 138, 139, 140, 145, 167
Crystal, 61, 68, 74, 77, 85, 86, 89, 106, 114, 121, 122, 125, 134, 135, 136, 144, 146, 152, 157, 161, 167, 168

D

Data Memory, 1, 2, 3, 6, 8, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23, 25, 38, 39, 42, 45, 46, 51, 53, 97, 101, 124, 126, 129, 130, 131, 143
Data Pointer (DPTR), 6, 45, 46, 84, 88, 124, 130, 147, 157, 159, 160, 161, 162, 163, 164, 169, 172, 173
Data Retention, 1, 3, 49, 56, 57, 58, 59, 61, 63, 64
Data Retention Current, 49, 56, 57, 58
Data Sheets, 144, 145
DS1215, 146, 157, 166
DS1283, 19, 146, 152, 155, 156, 161, 165
DS2250, 2, 3, 4, 5, 10, 13, 16, 39, 49, 59, 70, 80, 131, 146, 157, 166, 167
DS2251, 2, 3, 5, 10, 16, 19, 53, 54, 59, 80, 87, 131, 146, 152, 153, 155, 156, 161, 167
DS2252, 2, 3, 5, 10, 16, 19, 53, 55, 59, 79, 80, 146, 152, 153, 156, 161
DS5000, 1, 2, 3, 4, 8, 10, 12, 13, 14, 15, 16, 17, 20, 23, 24, 25, 27, 28, 38, 49, 50, 59, 60, 67, 68, 70, 73, 74, 75, 76, 77, 78, 79, 80, 81, 84, 86, 87, 88, 102, 123, 131, 132, 134, 137, 139, 140, 144, 146, 149, 157, 166
DS5000FP, 2, 3, 4, 10, 13, 49, 50, 70, 77, 80, 131, 146, 152, 167, 168
DS5000TK, 167
DS5001, 3, 10, 12, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 27, 29, 37, 39, 42, 43, 44, 51, 53, 56, 59, 67, 68, 70, 72, 73, 74, 80, 84, 87, 88, 97, 101, 102, 103, 104, 123, 131, 132, 133, 137, 139, 140, 144, 145, 152, 167, 168
DS5001FP, 2, 3, 4, 10, 16, 42, 43, 51, 52, 53, 58, 59, 71, 152
DS5002, 2, 8, 53, 56, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 168
DS5002FP, 2, 3, 4, 10, 16, 53, 59, 79, 80, 131, 146, 152
Dummy Bus Access, 73, 78, 80

E

ECE2, 13, 14, 16, 28, 38, 39, 87, 139, 146, 147, 166, 167
Encryption, 2, 3, 8, 73, 74, 75, 76, 77, 78, 79, 80, 81, 86, 139
Encryption Key, 3, 8, 73, 74, 76, 77, 79, 80, 84, 86, 139, 143
EPFW, 28, 30, 31, 62, 64, 65, 88, 90, 91, 92, 123
Error Messages, 74, 138, 140, 141
ESD, 60, 70
EWT, 28, 30, 31, 67, 68, 69, 70, 83, 86, 89, 123
EXBS, 20, 22, 42, 105
Expanded Bus, 6, 8, 12, 13, 14, 15, 17, 19, 20, 22, 42, 74, 99, 126, 127, 128, 129, 130, 147, 167
Expanded Memory, 97, 130
External Interrupts, 8, 35, 36, 61, 88, 90, 91, 94, 95, 97, 106
External Reset, 62, 83, 84, 86

F

Firmware Security, 73
Freshness Seal, 59

I

I/O, 1, 2, 3, 6, 8, 16, 34, 35, 43, 56, 60, 61, 62, 66, 80, 89, 97, 101, 105, 125, 127, 131, 140
Idle Mode, 31, 61, 62
IE, 28, 32, 35, 84, 90, 91, 93, 94, 95, 117, 121, 122, 123, 124, 158, 161
In–System Loading, 1, 168
Instruction Set, 2, 12, 45, 101, 169
Intel Hex File Format, 134, 137, 138, 139, 141
Internal Registers, 10, 11, 66, 126
Interrupt Acknowledge, 95, 96
Interrupt Priority, 36, 94
Interrupt Request Sources, 92
Interrupt Sources, 90, 95
Interrupt Vector, 8, 79, 80, 90, 91, 92, 96, 111
Interrupts, 2, 35, 36, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 102, 106, 111, 121, 122, 124, 147, 156, 166
IP, 36, 67, 68, 69, 70, 84, 86, 89, 90, 94, 95, 121, 122, 123, 124

K

Key, 3, 8, 23, 63, 66, 76, 77, 79, 80, 81, 82, 137, 138, 139, 143

L

Lifetime, 49, 56, 57, 58, 59, 80, 155, 166
Lithium Backed Circuits, 1, 22, 56, 58
Lithium Backup, 1, 56, 57, 59, 168
Lithium Cell, 1, 56, 57, 58, 59, 60, 63, 64, 166, 168
Loader ROM, 9
Lock, 3, 8, 14, 74, 76, 137, 167

M

MCON, 8, 12, 13, 15, 16, 17, 19, 20, 21, 23, 24, 25, 26, 28, 38, 39, 40, 67, 74, 84, 88, 123, 124, 139, 140, 143, 146, 153, 157, 159, 160, 161, 164, 165
Memory Interconnect, 49, 50, 51, 52, 53
Memory Map, 3, 6, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20, 21, 25, 27, 40, 73, 87, 88, 97, 126, 131, 146, 147, 152, 153, 167
Memory Map Control, 12, 14, 15, 16, 20, 22, 23
Memory Organization, 10, 13, 16
MSEL, 18, 19, 51, 52, 53

N

Negative Voltage, 60, 166, 168
No V_{LI}, 15, 16, 21, 22, 31, 37, 38, 39, 40, 43, 57, 70, 71, 83, 84, 86
Non–partitionable Mode, 16, 18, 19, 25

O

Oscillator, 9, 61, 62, 64, 68, 77, 85, 86, 99, 106, 111, 112, 113, 114, 117, 118, 125, 126, 128, 130, 132, 144, 150, 155, 166, 167

P

PA3–0, 15, 16, 17, 21, 67, 68, 86
PAA, 15, 16, 23, 24, 25, 28, 38, 39, 67, 68, 88, 139
Parallel I/O, 8, 97, 99, 132, 144, 145
Parallel Programming, 131, 144
Partition, 1, 2, 3, 8, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 38, 39, 40, 42, 51, 52, 67, 68, 71, 86, 87, 88, 123, 124, 127, 130, 139, 140, 143, 167
Partitionable Mode, 17, 25, 51, 167
PCON, 8, 27, 28, 30, 31, 61, 62, 64, 67, 68, 69, 70, 83, 84, 86, 88, 89, 90, 91, 113, 118, 122, 123, 124, 157, 158, 161, 166
Peripheral Enables, 3, 19, 20, 21, 40
Peripherals, 3, 10, 16, 19, 20, 21, 40, 76, 101, 111, 114, 140, 152, 153
PES, 19, 20, 21, 39, 40, 87, 153, 161, 164
PFW, 28, 30, 35, 61, 64, 65, 90, 91, 92, 93, 94, 95, 123
PM, 16, 17, 18, 20, 21, 22, 39, 40, 51, 71, 151
POR, 29, 30, 61, 64, 65, 67, 68, 83, 85, 86, 87, 89, 123
Port 0, 1, 6, 8, 12, 13, 19, 22, 42, 43, 45, 50, 51, 52, 53, 54, 55, 72, 74, 97, 99, 100, 101, 102, 105, 127, 128, 130, 139, 140, 167
Port 1, 50, 51, 52, 53, 54, 55, 72, 97, 98, 99, 100, 139, 140, 143
Port 2, 1, 6, 8, 12, 13, 17, 19, 22, 42, 43, 50, 51, 52, 53, 54, 55, 74, 97, 98, 99, 100, 101, 102, 105, 127, 130, 139, 140, 143, 167
Port 3, 50, 51, 52, 53, 54, 55, 97, 99, 100, 139, 140
Port1, 99
Power Management, 61, 65
Power On Reset Timing, 85
Power–Fail Interrupt, 1, 27, 31, 62, 64, 65, 90, 91
Power–Fail Warning, 8, 30, 61, 63, 64, 65, 88, 90, 91, 94, 95
Power–On Reset, 8, 9, 15, 16, 30, 31, 38, 39, 40, 61, 62, 64, 67, 68, 70, 83, 84, 85, 86, 89, 143, 144, 168
Product Description, 2
Program Loading, 1, 51, 76, 131, 132, 144
Program Memory, 8, 10, 12, 13, 14, 17, 22, 23, 25, 43, 46, 53, 62, 76, 79, 87, 126, 127, 128, 129, 130, 132, 139, 143
Program Status Flags, 47
Programmable Timers, 6, 33, 106, 108, 109
Programming, 1, 10, 12, 13, 16, 39, 45, 101, 127, 131, 132, 134, 143, 144
Programming Modes, 16, 144
PSW, 6, 12, 28, 41, 45, 47, 84, 112

R

RAM, 1, 2, 3, 6, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 23, 24, 26, 45, 49, 50, 51, 53, 56, 58, 61, 63, 64, 71, 73, 74, 76, 77, 78, 79, 80, 81, 86, 88, 91, 121, 123, 126, 127, 128, 131, 134, 137, 138, 139, 140, 141, 143, 145, 152, 157, 161, 166, 167

Random Number, 42, 73, 77, 79, 80, 81
Range, 12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 37, 38, 40, 43, 46, 51, 67, 70, 71, 76, 86, 87, 127, 130, 138, 139, 140, 141, 143, 167
Real–Time Clock, 1, 2, 3, 16, 19, 49, 58, 59, 76, 87, 140, 146, 154, 155, 157, 161
Real–Time Clock Command Register, 155
Real–Time Clock Memory Map, 154
Recommended SRAM, 49
Reset, 1, 2, 3, 8, 9, 15, 16, 21, 22, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 47, 56, 57, 61, 62, 64, 65, 66, 67, 68, 69, 70, 71, 72, 79, 80, 83, 84, 85, 86, 87, 88, 89, 90, 91, 93, 94, 97, 104, 105, 106, 107, 108, 112, 113, 117, 124, 131, 132, 137, 140, 143, 144, 145, 147, 149, 155, 157, 161, 166, 167, 168
Reset Conditions, 64, 69, 70, 83, 84, 86, 91
Reset Sources, 83
Reset States, 61, 87
Reset Vector, 87, 132, 166
RG0, 17, 18, 21, 22, 40, 43
RG1, 17, 18, 21, 39, 40
RNR, 22, 42, 79, 84, 105
ROM, 1, 3, 6, 8, 9, 10, 11, 12, 73, 74, 131, 144, 167
RPC, 42, 43, 44, 101, 102, 103, 104, 105, 131, 132, 133, 140, 144, 145
RPC Program Mode, 144
RPCTL, 12, 16, 17, 20, 21, 22, 25, 40, 42, 43, 67, 79, 84, 101, 104, 105, 139, 140
RTC, 28, 36, 58, 67, 68, 69, 70, 86, 87, 89, 122, 146, 147, 149, 150, 152, 153, 155, 156, 157, 159, 161

S

SCON, 6, 28, 34, 35, 84, 89, 90, 91, 111, 112, 113, 114, 115, 118, 121, 122, 123, 124, 158, 161
Scratchpad Registers, 6, 10, 11, 63
SDI, 55, 79, 80, 82
Security, 1, 2, 3, 8, 14, 15, 16, 53, 73, 74, 77, 78, 79, 80, 81, 82, 131, 137, 143, 146, 152
Security Lock, 8, 14, 15, 16, 21, 22, 38, 39, 40, 43, 73, 74, 76, 77, 79, 80, 86, 127, 139, 143
Selection Guide, 4
Self Destruct, 2, 3, 73, 79, 80
Serial Communication, 90, 91, 121, 134, 167, 168
Serial I/O, 6, 8, 34, 35, 89, 90, 93, 94, 95, 111, 112, 113, 114, 117, 118, 121, 123, 157
Serial Interface, 134
Serial Interrupt, 35, 90, 91, 93, 114, 115, 117, 118, 121, 122, 124
Serial Port, 1, 9, 22, 23, 25, 30, 35, 36, 61, 87, 88, 89, 90, 91, 93, 94, 97, 110, 111, 112, 114, 119, 121, 122, 123, 124, 131, 132, 133, 135, 140, 143, 144, 157, 158, 161, 167, 168
Serial Programming, 134
SMOD, 28, 30, 113, 114, 118, 122, 123, 124
Soft Reload, 22, 23, 25
Software Encryption, 8, 75, 86, 143
Software Security, 1
Special Function Register Map, 6, 28, 29
Special Function Registers, 6, 8, 10, 12, 20, 27, 45, 62, 63, 64, 66, 68, 83, 84, 85, 86, 97, 101, 106, 140
SRAM, 1, 2, 3, 8, 10, 11, 12, 13, 16, 17, 19, 49, 50, 51, 52, 53, 56, 58, 59, 153
Stop Mode, 8, 9, 27, 31, 61, 62, 67, 68, 86, 166, 167

T

TA, 24, 25, 26, 67, 69, 88, 89, 124, 157, 161
TCON, 28, 107
Timed Access, 8, 15, 16, 21, 22, 23, 24, 25, 26, 27, 30, 31, 36, 38, 39, 42, 61, 62, 66, 67, 68, 69, 70, 83, 86, 124, 166
Timer Interrupts, 81, 90, 91, 95, 122, 155
Timers, 2, 6, 8, 32, 33, 42, 61, 66, 68, 69, 70, 81, 86, 89, 91, 105, 106, 107, 108, 109, 110, 111, 113, 114, 117, 118, 121, 122, 123, 124, 145, 152, 155, 157, 161, 166, 167
Timing
  Expanded Data Memory, 130
  Expanded Program Memory, 127
  Instruction, 126, 127
  Power–on Reset, 85
TMOD, 28, 33, 84, 89, 106, 107, 110, 113, 121, 123, 124, 158, 161

U

UART, 2, 3, 19, 37, 71, 111, 112, 121, 122, 133, 134, 145
Unlock, 3, 74, 76, 77, 79, 80, 137, 139

V

Vector RAM, 3, 8, 73, 74, 77, 79, 80, 81, 139

W

Watchdog, 1, 2, 3, 8, 9, 21, 22, 27, 31, 36, 37, 39, 40, 43, 61, 62, 66, 67, 68, 69, 71, 83, 84, 86, 87, 89, 152, 155, 156, 166, 167, 168
Watchdog Timer Reset, 30, 67, 68, 70, 83, 86
WTR, 28, 29, 30, 68, 69, 70, 83, 86, 123

Random Number, 42, 79, 77, 79, 80, 81
Range, 12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 27,
38, 40, 46, 51, 67, 70, 71, 76, 86, 87, 127,
130, 139, 139, 140, 141, 143, 167
Real–Time Clock, 1, 2, 3, 16, 19, 40, 59, 69, 78, 87,
140, 146, 154, 155, 167, 161
Real–Time Clock Command Register, 155
Real–Time Clock Memory Map, 154
Recommended SRAM, 49
Reset, 1, 2, 3, 8, 9, 15, 16, 21, 22, 30, 31, 32, 33, 34,
36, 38, 67, 38, 39, 40, 41, 42, 46, 47, 58, 67, 81,
82, 84, 85, 86, 87, 88, 89, 70, 71, 72, 78, 80, 83,
84, 85, 86, 87, 88, 89, 90, 91, 93, 94, 97, 104,
105, 106, 107, 108, 112, 113, 117, 124, 131, 132,
137, 140, 143, 144, 145, 147, 149, 155, 157, 161,
166, 167, 168
Reset Conditions, 64, 69, 70, 83, 84, 86, 91
Reset Sources, 69
Reset States, 61, 67
Reset Vector, 67, 132, 155
RGO, 17, 18, 21, 22, 40, 43
RGT, 17, 18, 21, 39, 40
RNR, 22, 42, 79, 84, 105
ROM, 1, 3, 6, 8, 9, 10, 11, 12, 73, 74, 131, 144, 167
RPC, 42, 43, 101, 102, 103, 104, 105, 131, 132,
139, 140, 144, 145
RPC Program Mode, 144
RPCTL, 12, 16, 17, 20, 21, 22, 25, 40, 42, 43, 67, 79,
94, 101, 104, 105, 139, 140
RTC, 29, 56, 58, 67, 68, 69, 70, 86, 87, 89, 122, 146,
147, 149, 150, 152, 153, 155, 156, 157, 159, 161

S

SCON, 6, 23, 34, 35, 64, 89, 90, 91, 111, 112, 113,
114, 115, 119, 121, 122, 123, 124, 156, 161
Scratchpad Registers, 6, 10, 11, 63
SDI, 55, 79, 80, 82
Security, 1, 2, 3, 8, 14, 15, 16, 53, 73, 74, 77, 78, 79,
80, 81, 82, 131, 137, 143, 146, 152
Security Lock, 8, 14, 15, 16, 21, 22, 36, 39, 40, 43,
73, 74, 75, 77, 79, 80, 96, 127, 130, 143
Selection Guide, 4
Self Destruct, 2, 3, 73, 79, 80
Serial Communication, 30, 91, 121, 124, 167, 168
Serial I/O, 6, 8, 34, 35, 89, 90, 93, 94, 95, 111, 112,
113, 114, 117, 118, 121, 123, 157
Serial Interface, 124
Serial Interrupt, 35, 90, 91, 93, 114, 115, 117, 118,
121, 122, 124
Serial Port, 1, 8, 22, 23, 25, 30, 35, 36, 61, 67, 88, 89,
90, 91, 93, 94, 97, 110, 111, 112, 114, 119, 121,
122, 123, 124, 131, 132, 153, 160, 163, 164,
167, 168, 161, 167, 168

Serial Programming, 134
SMOD, 28, 30, 113, 114, 119, 122, 123, 124
Soft Reload, 22, 23, 25
Software Encryption, 8, 75, 86, 143
Software Security, 1
Special Function Register Map, 6, 28, 29
Special Function Registers, 6, 8, 10, 12, 20, 27, 46,
62, 63, 64, 66, 68, 69, 83, 84, 86, 88, 97, 101, 106,
146
SRAM, 1, 2, 3, 8, 10, 11, 12, 13, 16, 17, 18, 19, 46, 50,
51, 52, 55, 56, 58, 59, 89, 153
Stop Mode, 3, 6, 27, 31, 61, 62, 67, 68, 86, 166, 167

T

TA, 24, 25, 26, 67, 69, 88, 89, 124, 157, 161
TCON, 29, 107
Timed Access, 8, 15, 16, 21, 22, 23, 24, 25, 26, 27,
30, 31, 36, 38, 39, 42, 81, 82, 86, 87, 88, 89, 70,
88, 86, 124, 166
Timer Interrupts, 61, 90, 91, 95, 122, 153
Timers, 2, 6, 8, 22, 23, 38, 42, 61, 86, 88, 89, 70, 81, 86,
89, 91, 106, 108, 107, 108, 109, 110, 111, 113,
114, 117, 118, 121, 122, 123, 124, 145, 152, 156,
157, 161, 166, 167
Timing
Expanded Data Memory, 130
Expanded Program Memory, 127
Instruction, 128, 127
Power–on Reset, 85
TMOD, 28, 33, 84, 89, 105, 107, 110, 113, 121, 123,
124, 156, 161

U

UART, 2, 3, 19, 37, 71, 111, 112, 121, 122, 133, 154,
146
Unlock, 3, 74, 76, 77, 79, 80, 127, 159

V

Vector RAM, 3, 8, 73, 74, 77, 79, 80, 81, 139

W

Watchdog, 1, 2, 3, 8, 9, 21, 22, 27, 31, 36, 37, 39, 40,
42, 61, 62, 66, 67, 68, 69, 71, 83, 84, 86, 87, 89,
152, 165, 156, 166, 167, 183
Watchdog Timer Reset, 30, 67, 38, 70, 83, 86
WTR, 28, 29, 30, 66, 69, 70, 83, 86, 123